Catalogue formations 2019

Ateliers
Classes virtuelles
Accompagnement projets
Formations inter-entreprises

Pythagore F.D.

Apprendre à apprendre

Pythagore F.D. : Apprendre à Apprendre

Nouveautés 2019 :

Parce que la maîtrise des technologies BigData nécessite des compétences multiples et une vision d'ensemble des outils disponibles, nous vous proposons des **cycles certifiants pour Data-scientists** et **pour architectes BigData**, permettant d'acquérir les principes fondamentaux du calcul distribué, des bases de données non structurées, ainsi que l'expertise adéquate pour la mise en oeuvre d'architectures **Hadoop**, ou de solutions de traitement de flux de données avec **Kafka**, **Storm** et de **Machine Learning**.

Parmi les nouveautés techniques, nous proposons des formations sur la **programmation Go**, la supervision avec **Icinga**, le calcul distribué avec **Hadoop EMR.**

Nos méthodes pédagogiques :

Apprendre, concevoir, intégrer ... le meilleur moyen de comprendre les nouveaux concepts et les technologies est la mise en pratique.

Le matériel pédagogique mis à disposition pour chaque formation permet de mettre en pratique tous les concepts abordés : clusters de calcul pour MapReduce, baies de stockage distribué pour cassandra, accès Amazon pour la gestion de ressources dans le cloud, etc ...

Nos formations sont disponibles en stages inter-entreprises, en intra dans notre centre de formation ou sur site, et en classes virtuelles.

Apprendre, concevoir, intégrer ...

Nos domaines d'expertise :

- les bases de données et le BigData avec NoSQL, Cassandra, MongoDB, Hadoop, Spark, Storm ...
- la virtualisation et l'orchestration avec xen, kvm, lxc, Docker, et le cloud : cloudstack et openstack, openNebula, cobbler, etc ...
- TCP/IP (IPv6, snmp, Architecture, Sécurité, Administration de réseaux IP, VoIP, ...)
- Unix et Linux, et les applicatifs Apache, Openldap, Squid, Nagios, Zabbix, OCS/GLPI, puppet , chef...
- Développement (langage C, Java, Perl, Python, Go ...)

Notre centre de formation :

Sur le plan pratique, notre centre de formation est situé 11, rue du Faubourg Poissonnière, à Paris (9è), à deux pas des Grands Boulevards, à proximité de nombreux restaurants, hôtels, etc ...

01 55 33 52 10
www.pythagore-fd.fr

Pythagore F.D.
Apprendre à apprendre

Filières BigData

BigData concepts et enjeux
BigData Architecture et technologies

Introduction à NoSQL
Stockage distribué avec Ceph
BigData avec Cassandra
NoSQL avec mongoDB
Neo4J Graphes et analyse

Indexation avec Elasticsearch
Elasticsearch programmation
Elasticsearch : infrastructure et administration

Hadoop Ecosystème
Administration Hadoop HortonWorks
Administration Hadoop Cloudera
Hadoop Développement
Hadoop : stockage avec HBase
Hadoop : Infrastructure sécurisée
Hadoop : Analytics

Spark Mise en oeuvre et programmation8
Flux de données avec Storm
Kafka : ingestion de données

Pig Apache : développement de scripts
Data Classification et Machine Learning
Programmation Scala
Dataviz : solutions opensource
Environnement R : traitements statistiques
Talend OpenStudio : intégration de données
Talend OpenStudio : optimisation

Cursus certifiant Data Scientist
Cursus certifiant architecte BigData

Objets connectés:des OS embarqués au cloud

CB100

Durée: 1 jour
680 €

13 février — 4 septembre
23 mai — 29 novembre

Public:
Décideurs,architectes,chefs de projet et toute personne souhaitant aborder les technologies des objets connectés.

Objectifs:
Comprendre quelles sont les briques technologiques mises en oeuvre dans les objets connectés :depuis les systèmes embarqués jusqu'au stockage des données en passant par les technologies réseaux utilisées.Ce cours est illustré de nombreuses démonstrations et travaux pratiques.

Connaissances préalables nécessaires:
Connaissance générale des systèmes d'informations.

Programme:

Introduction : Définitions,applications,services : domotique, santé, loisirs. L'internet des objets. Les acteurs et produits du marché

Bases embarquées : Plate-formes matérielles (Intel,Samsung) et logicielles. Les systèmes classiques android, IOS, systèmes embarqués : UI, Brillo, LiteOS
Kit de développement Galileo.Les modules Arduino.

Communications : Protocoles: mqtt, bluetooth, wifi, 3G/4G, etc ... Avec un autre objet : M2M, Avec le réseau internet : vers un serveur, vers le cloud.Les plate-formes IoT, définition de standards : OpenInterconnect Consortium. Mise en évidence avec AWS/IoT en mqtt sur websocket.

Traitement des données : Types de données collectées.Données locales, limites. Transfert et stockage sur une autre machine, dans le cloud ou sur internet. Analyse des données et fourniture de services associés

Sécurité : Techniques de hacking des objets connectés. Protection par vpn. Authentification et autorisations. Protection des données, législation

CB200

Blockchain : principes et technologies

Durée: *1 jour*
680 €

14 février
21 juin 4 octobre

Public:
Décideurs,architectes,chefs de projet et toute personne souhaitant comprendre le principe de la blockchain, les applications, et la mise en oeuvre.

Objectifs:
Comprendre les principe, les apports, les mécanismes mise en oeuvre dans le cadre de la blockchain.

Connaissances préalables nécessaires:
Connaissance générale des systèmes d'informations.

Programme:

Introduction : Principe, historique,
notions de transactions, de blocs, de stockage distribué, de noeuds du réseau, de mineurs,
exemples de blockchain, cas d'usage concrets et applications:
crypto-monnaies, smart contracts, traçabilité, ...

Cas du bitcoin : Fonctionnement,
Description des transactions, comptes, wallet
Principe du minage : assemblage des transactions en blocs,
présentation de la structure des blocs et du calcul de l'empreinte
Exemple de mise en oeuvre technique :outils matériels et logiciels pour configurer un noeud
Démonstrations sur une plate-forme Linux

Blockchain Ethereum : Plate-forme de smart-contracts,
fonctionnement, outils,démonstrations de minage sur Linux
Choix du mode de validation : preuve de travail ou preuve de participation

Limites des blockchains : Problème de la consommation énergétique.
Risque de prise de contrôle par un groupe de mineurs,
Défauts de sécurité

BigData : concepts et enjeux

CB000

Durée: 1 jour
670 €

21 janvier
1er avril
24 juin
23 septembre
18 novembre

Public:
Chefs de projets, architectes, data-scientists, et toute personne souhaitant comprendre les impacts du BigData sur l'entreprise au niveau du traitement des données, des architectures, de l'organisation.

Objectifs:
Comprendre les concepts et les apports du BigData, les impacts sur l'organisation de l'entreprise.

Connaissances préalables nécessaires:
Il est demandé aux participants d'avoir une bonne culture générale sur les systèmes d'information.

Programme:
Introduction : A l'origine du BigData : traitement de volumes importants de données non structurées,
traitements optimisés de flux de données au fil de l'eau,
liés aux nouvelles technologies et aux nouveaux usages.
Domaines concernés : recherche scientifique, médical, e-commerce, sécurité, prédictif, ...
Exemples : lutte contre la criminalité, fraude, santé, ressources énergétiques
Apports des évolutions techniques sur différents aspects : stockage, indexation/recherche, calcul.
Concepts clés : ETL, Extract Transform Load, CAP, 3V, 4V, données non structurées, prédictif, Machine Learning.
Quelques applications : Watson (IBM), Amazon Rekognition
Le positionnement des technologies de cloud, BigData et noSQL, de data-mining. Eléments d'architecture.
Gouvernance des données :
importance de la qualité des données, fiabilité, durée de validité, sécurité des données
Aspects législatifs : sur le stockage, la conservation de données, etc ...
sur les traitements, la commercialisation des données, des résultats

BigData : concepts et enjeux

CB000

Stockage distribué	: Caractéristiques NoSQL Les différents modes et formats de stockage. Besoin de distribution. Définition de la notion d'élasticité. Principe du stockage réparti : Définitions : réplication, sharding, gossip protocol, hachage, Systèmes de fichiers distribués : GFS, HDFS, Ceph Les bases de données : Cassandra, HBase, MongoDB, CouchBase, Riak, BigTable, ..
Calcul et restitution.	: Apport des outils de calculs statistiques Langages adaptés aux statistiques, liens avec les outils BigData. Outils de calcul et visualisation : R, SAS, Spark, Tableau, QlikView, ... Caractéristiques et points forts des différentes solutions.
Evolutions	: Liens vers les nouveaux métiers : Hadoop scientists, Data scientists, CDO. Analyse des données au service de l'entreprise Rôle de la DSI dans la démarche BigData. Ouverture sur l'OpenData : principe, la démarche publique, les licences. Exemple : portail data.gouv.fr Les offres Saas BigData comme Google BigQuery. Les limites. Les nouveautés annoncées.

BigData Architecture et technologies

CB001

Durée: 2 jours
1200 €

22 au 23 janvier
2 au 3 avril
25 au 26 juin
24 au 25 septembre
19 au 20 novembre

Public:

Chefs de projets, architectes, développeurs, data-scientists, et toute personne souhaitant connaître les outils et solutions pour concevoir et mettre en oeuvre une architecture BigData.

Objectifs:

Comprendre les concepts essentiels du BigData, et les technologies implémentées. Savoir analyser les difficultés propres à un projet BigData, les freins, les apports, tant sur les aspects techniques que sur les points liés à la gestion du projet.

Connaissances préalables nécessaires:

Il est demandé aux participants d'avoir une bonne culture générale sur les systèmes d'information.

Programme:

Introduction : L'essentiel du BigData : calcul distribué, données non structurées. Besoins fonctionnels et caractéristiques techniques des projets.La valorisation des données.Le positionnement respectif des technologies de cloud, BigData et noSQL, et les liens, implications.
Concepts clés : ETL, Extract Transform Load, CAP, 3V, 4V, données non structurées, prédictif, Machine Learning.
Quelques applications : Watson (IBM), Amazon Rekognition.
L'écosystème du BigData : les acteurs, les produits, état de l'art.Cycle de vie des projets BigData.
Emergence de nouveaux métiers : Datascientists, Data labs, Hadoop scientists, CDO, ...
Rôle de la DSI dans la démarche BigData. Gouvernance des données:importance de la qualité des données, fiabilité, durée de validité, sécurité des données
Aspects législatifs : sur le stockage, la conservation de données, etc ...sur les traitements, la commercialisation des données, des résultats

BigData Architecture et technologies

CB001

Stockage distribué	: Caractéristiques NoSQL.Les différents modes et formats de stockage. Les types de bases de données : clé/valeur, document, colonne, graphe.Besoin de distribution. Définition de la notion d'élasticité.Principe du stockage réparti. Définitions : réplication, sharding, gossip protocol, hachage, Systèmes de fichiers distribués : GFS, HDFS, Ceph.Les bases de données : Cassandra, DynamoDB, Accumulo, HBase, MongoDB, CouchBase, Riak, BigTable, .. Caractéristiques NoSQL :structure de données proches des utilisateurs, développeurs. Les types de bases de données : clé/valeur, document, colonne, graphe. Données structurées et non structurées, documents, images, fichiers XML, JSON, CSV, ... Les différents modes et formats de stockage. Stockage réparti : réplication, sharding, gossip protocol, hachage Systèmes de fichiers distribués : GFS, HDFS.Quelques exemples de produits et leurs caractéristiques : Cassandra, MongoDB, CouchDB, DynamoDB, Riak, Hadoop, HBase, BigTable, ... Qualité des données, gouvernance de données.
Indexation et recherche	: Moteurs de recherche.Principe de fonctionnement. Méthodes d'indexation. Mise en oeuvre avec elasticsearch.Exemple de Lucene/solr.Recherche dans les bases de volumes importants. Exemples de produits et comparaison : Dremel, Drill, ElasticSearch, MapReduce,
Calcul et restitution, intégration	: Différentes solutions : calculs en mode batch, ou en temps réel,sur des flux de données ou des données statiques. Les produits : langage de calculs statistiques, R Statistics Language, sas, RStudio;outils de visualisation : Tableau, QlikView Ponts entre les outils statistiques et les bases BigData.Outils de calcul sur des volumes importants : storm en temps réel, hadoop en mode batch. Zoom sur Hadoop : complémentarité de HDFS et MapReduce.Restitution et analyse : logstash, kibana, elk, pentaho Présentation de pig pour la conception de tâches MapReduce sur une grappe Hadoop.

Introduction à NoSQL

CB002

Durée: 1 jour
670 €

24 janvier
4 avril
27 juin
26 septembre
21 novembre

Public:

Experts en bases de données, chefs de projet et toute personne souhaitant comprendre le fonctionnement et les apports des bases NoSQL.

Objectifs:

Connaître les caractéristiques techniques des bases de données NoSQL, les différentes solutions disponibles. Identifier les critères de choix.

Connaissances préalables nécessaires:

Connaissance générale des systèmes d'informations et des bases de données.

Programme:

Introduction : origine des bases de données, les notions de transaction, les SGBD, la standardisation SQL,
l'arrivée de nouveaux besoins : volumes importants liés aux technologies et aux nouveaux usages,
traitements optimisés de flux de données au fil de l'eau.
Développement des techniques sur différents aspects : stockage, indexation/recherche, calcul.
Définition ETL : Extract Transform Load.

Caractéristiques NoSQL : Structure de données proches des utilisateurs, développeurs: sérialisation, tables de hachage, JSON.
Priorité au traitement du côté client.
Protocoles d'accès aux données, interfaces depuis les langages classiques.
Données structurées et non structurées, documents, images,
Stockage réparti : réplication, sharding, protocole gossip, hachage,..
Parallélisation des traitements : implémentation de MapReduce.
Cohérence des données et gestion des accès concurrents : "eventual consistancy" et multi-version concurrency control.

Introduction à NoSQL

CB002

Principaux acteurs	: Les solutions NoSQL et leurs choix techniques : CouchDB, MongoDB, Cassandra, HBase (Hadoop), ElasticSearch, .. Démonstrations avec Cassandra et couchDB. Critères de choix.
Mise en oeuvre	: Points à vérifier : méthode d'utilisation des données, format de stockage JSON, XML, choix de la clé, notion de clé composite, ... aspects matériels, besoins en mémoire, disques, répartition, .. Import des données : outils et méthodes selon les moteurs NoSQL

Stockage distribué avec Ceph

CB004

Durée: 2 jours
1200 €

31 janvier au 1er février
25 au 26 avril
19 au 20 septembre
28 au 29 novembre

Public:

Administrateurs systèmes Linux souhaitant utiliser un système de stockage distribué.

Objectifs:

Comprendre le fonctionnement de Ceph, savoir le mettre en oeuvre et le configurer.

Connaissances préalables nécessaires:

Connaissance de l'administration des systèmes Linux.

Programme:

Introduction : Système de stockage distribué,
Historique et intégration dans le noyau Linux. Fonctionnalités et points forts.
Stockage objet, stockage de blocs, système de fichiers CephFS. Interfaçage avec des plate-formes de cloud

Mise en oeuvre : Architecture : Ceph Monitor, services OSD, Ceph Metadata Server
Configuration d'un cluster de stockage Ceph multi-noeuds
Gestion des noeuds, des serveurs de stockage, des Ceph Monitors

Clients Ceph : Différents clients Ceph selon la fonctionnalité utilisée.
Stockage de blocs : installation Ceph et configuration d'un bloc et montage depuis un poste client
Système de fichiers Ceph : création d'un système de fichier, clé d'authentification, montage du file system
Stockage objet : installation et création d'une instance Ceph Object Gateway
APIs de stockage objet Ceph : compatibilité AWS S3, et OpenStack Swift

Sécurité et fiabilité : Architecture haute disponibilité.Installation et configuration de ceph-mgr (Ceph Manager Daemon).Tests de fiabilité.

BigData avec Cassandra

CB010

Durée: 3 jours
1760 €

4 au 6 février
13 au 15 mai
9 au 11 septembre
25 au 27 novembre

Public:
Chefs de projet, gestionnaires de bases de données.

Objectifs:
Connaître les apports de Cassandra, savoir l'installer et le configurer et maîtriser le CQL. Gérer Cassandra en production avec OpsCenter. Savoir interfacer avec Hadoop, Spark.

Connaissances préalables nécessaires:
Connaissances générales sur les bases de données.

Programme:

Introduction : Historique, fonctionnalités de Cassandra, licence
Format des données,"key-value", traitement de volumes importants,
haute disponibilité, système réparti de base de données, ...

Installation et configuration : Prérequis. Plate-formes supportées. Etude du fichier de configuration : conf/cassandra.yaml
Répertoire de travail, de stockage des données, gestion de la mémoire.
Démarrage d'un noeud et test de l'interface cliente cqlsh.

CQL : Commandes de base : connexion au système de base de données,
création de colonnes,insertion, modification recherche,
Le CQL : Cassandra Query Language. Exécution de scripts.
Comment écrire des requêtes? Approches.

Gestion de la grappe : Principe.Préparation du premier noeud : adresse d'écoute.
Configuration de nouveaux noeuds.Notion de bootstrapping et de token.
Paramètres listen_address et rpc_address.
Réplication : topologie du réseau et EndpointSnitch.Stratégie de réplication. Ajout de noeuds, suppression.
Cassandra dans un cloud. Mise en oeuvre avec OpenStack.

BigData avec Cassandra

CB010

Supervision	: OpsCenter : installation, lancement. Utilisation de base. Supervision avec nodetool cfstats, ou export JMX vers des outils de supervision comme Nagios.
Exploitation	: Sauvegardes. Import/export au format JSON.
Support Hadoop	: Principe de MapReduce. Implémentation Hadoop. Mise en oeuvre depuis Cassandra.
Support Spark	: Description rapide de l'architecture spark. Mise en oeuvre depuis Cassandra. Execution de travaux Spark s'appuyant sur une grappe Cassandra.

Base de données NoSQL avec MongoDB

CB017

Durée: 3 jours
1760 €

4 au 6 mars
3 au 5 juin
7 au 9 octobre
16 au 18 décembre

Public:
Chefs de projet, gestionnaires de bases de données.

Objectifs:
Comprendre le fonctionnement de MongoDB, savoir l'installer, le configurer,l'administrer, créer des requêtes d'interrogation, et mettre en oeuvre la réplication.

Connaissances préalables nécessaires:
Connaissance des principes classiques des bases de données.

Programme:

Introduction : Présentation MongoDB, historique du projet, les versions
Structure des données :notions de documents, de collections
Le format BSON (Binary JSON), comparaison avec JSON
Fonctionnalités de MongoDB
Interfaces disponibles

Installation et configuration : Plate-formes supportées.
Packages nécessaires, scripts de lancement.
Travaux pratiques : installation, lancement du service mongod. Tests de connexion

Interpréteur : Présentation du shell Mongo.
Initialisation et premières requêtes.
Opérations CRUD : Create, Read, Update, Delete.
Importation, exportation de données.
Travaux pratiques : la méthode find, critères de requêtes, les types, les curseurs, ..

Sécurité : Mise en oeuvre de l'authentification dans MongoDB.
Paramètres de configuration auth et keyFile
Gestion des rôles.
Etude de la collection system.users

CB017

Base de données NoSQL avec MongoDB

Le sharding	: Définition, principe de fonctionnement. Exemples de mise en oeuvre du sharding, configuration et administration Réplication : principe des replica sets et mise en oeuvre, Mécanisme de fail-over automatique Partitionnement des données avec le sharding Optimisation : gestion des connexions, ajout de serveurs, équilibrage
Exploitation	: Gestion des opérations, analyse, points de blocage. Mise en oeuvre de mongotop et mongostat. Gestion des index, chargement des données en mémoire Analyse des logs
Administration	: Supervision : gestion de la mémoire, analyse des performances, tuning. Sauvegardes d'un serveur, de cluster Travaux pratiques avec mongodump.

Neo4J : graphes et analyse

CB018

Durée: 1 jour
660 €

22 mars — 11 octobre
5 juillet — 13 décembre

Public:
Chefs de projet, gestionnaires de bases de données.

Objectifs:
Comprendre le fonctionnement de Neo4j, savoir le mettre en oeuvre pour le stockage de données de type graphe.

Connaissances préalables nécessaires:
Connaissance des principes classiques des bases de données.

Programme:

Introduction : Présentation Neo4j, les différentes éditions, license
Fonctionnalités, stockage des données sous forme de graphes
CQL : Cypher Query Language
Positionnement par rapport aux autres bases de données, apports de Neo4j
L'analyse de données.

Installation et configuration : Travaux pratiques : installation de Neo4J Community Edition
Premiers pas avec l'interface web.
Création de données, requêtage
Import de données

Cypher Query Language : Syntaxe, description des relations avec CQL,les patterns
Les clauses d'écriture : set, delete, remove, foreach,
de lecture : match, optional match, where, count, ..
Les fonctions :
Travaux pratiques : création d'un graphe,
Requêtes de recherche, navigation dans le graphe

Drivers : Description des APIs disponibles:
.Net, Java, Javascript, Python

Exploitation : Sauvegardes et restaurations
Indexation

ElasticSearch : indexation

CB020

Durée: 1 jour
680 €

4 mars — 30 septembre
17 juin — 16 décembre

Public:
Chefs de projet, développeurs, architectes.

Objectifs:
Comprendre le fonctionnement et les apports d'Elasticsearch dans le traitement de données.

Connaissances préalables nécessaires:
Connaissances générales des systèmes d'informations

Programme:

Introduction : Présentation, fonctionnalités, licence
Définitions et techniques d'indexation
Positionnement Elasticsearch et les produits complémentaires : Watcher, Marvel, Kibana,X-Pack, Logstash, Beats
Principe : base technique Lucene et apports d'ElasticSearch

Installation de base : Prérequis techniques.
Utilisation de l'interface d'administration Marvel

Outils d'interrogation : Java API avec"Node client" et "Transport client"
Autres clients : Perl, Python, Ruby, etc...
Interface http, travaux pratiques, démonstration

Traitement des données : Structure des données. stockage, indexation
Format des données.
Conversion au format JSON des données à traiter.
Interrogations avec Search Lite et avec Query DSL (domain-specific language)
Notion de 'filtre' pour affiner des requêtes.

ElasticSearch : mise en oeuvre et programmation

CB021

Durée: 2 jours
1250 €

5 au 6 mars — 1er et 2 octobre
18 et 19 juin — 17 au 18 décembre

Public:
Architectes techniques, ingénieurs système, administrateurs..

Objectifs:
Comprendre le fonctionnement et les apports d'Elasticsearch dans le traitement de données, et savoir le mettre en oeuvre, analyser les données, programmer des requêtes et créer des rapports et tableaux de bord avec kibana.

Connaissances préalables nécessaires:
Connaissances générales des systèmes d'information, et des systèmes d'exploitation (Linux ou Windows). Les travaux pratiques sont réalisés sur Linux.Connaissance d'un langage de programmation structuré

Programme:

Introduction : Présentation ElasticSearch, fonctionnalités, licence
Les différentes versions : fonctionnalités et particularités des versions de 2.0 à 5.0.
Nouveautés de la version 6.0.
Positionnement d'Elasticsearch et des produits complémentaires : Watcher, Marvel, Kibana, Logstash, Beats, X-Pack
Principe : base technique Lucene et apports d'ElasticSearch.Fonctionnement distribué

Installation et configuration : Prérequis techniques.Utilisation de l'interface Marvel.Premiers pas dans la console Sense.

Format et stockage des données : Format des données. Conversion au format JSON des données à traiter.
Structure des données. Stockage, indexation. Terminologie Elasticsearch : notions de document, type, index.Métadonnées : _index, _type, _ID
Choix de l'identifiant par l'application avec l'API index, ou génération automatique d'un identifiant..
Indexation inversée.

ElasticSearch : mise en oeuvre et programmation

CB021

Outils d'interrogation	: Java API avec "Node client" et "Transport client". API RESTful en HTTP Exemples de requêtes simples et plus complexes : recherche de «phrases», extraction de plusieurs documents, etc .. Notion de pertinence du résultat : «score» Requêtes avec Search Lite et avec Query DSL (domain-specific language) Utilisation de 'filtre' pour affiner des requêtes. Autres clients : Perl, Python, Ruby, etc...Aggrégation de résultats.
Mises à jour	: Fonctionnement d'Elasticsearch pour les ajouts, modifications, suppression. Notion de version affectée par Elasticsearch. L'API bulk pour les traitements groupés. Réalisation de scripts avec groovy
Gestion des accès concurrents	: Utilisation du numéro de version.Gestion par l'application : différentes méthodes selon les contraintes fonctionnelles. Utilisation d'un numéro de version externe.
Kibana présentation	: Fonctionnalités : recherche, visualisation, création de tableaux de bord et graphiques à partir des données fournies par Elacticsearch
Kibana, mise en oeuvre	: Installation, configuration du mapping avec Elasticsearch.Paramétrage dans le fichier kibana.yml. Mapping automatique ou manuel.Configuration des indexes à explorer. Visualisation et sauvegarde de graphiques, étude des différents types de graphiques disponibles, création de tableaux de bord et rapports à partir des graphiques.

ElasticSearch : infrastructure et administration

CB022

Durée: 2 jours
1250 €

7 au 8 mars — 3 et 4 octobre
20 et 21 juin — 19 et 20 décembre

Public:
Architectes techniques, ingénieurs système, administrateurs..

Objectifs:
Comprendre le fonctionnement d'Elasticsearch, savoir l'installer et le configurer, gérer la sécurité avec X-Pack, et installer / configurer kibana pour le mapping sur les données Elasticsearch.

Connaissances préalables nécessaires:
Connaissances générales des systèmes d'information, et des systèmes d'exploitation (Linux ou Windows). Les travaux pratiques sont réalisés sur Linux.

Programme:

Introduction : Présentation ElasticSearch, fonctionnalités, licence
Positionnement d'Elasticsearch et des produits complémentaires : Watcher, Kibana, Logstash, Beats, X-Pack
Principe : base technique Lucene et apports d'ElasticSearch
Fonctionnement distribué

Installation et configuration : Prérequis techniques.
Installation depuis les RPM.
Utilisation de l'interface X-Pack monitoring.
Premiers pas dans la console Sense.
Etude du fichier : elasticsearch.yml

Kibana : Présentation : objectifs, collecte de données, logs, ... par les APIs d'administration et de supervision ;
Stockage dans elasticsearch et mise à disposition dans une interface web de graphiques
Démonstrations.

Clustering : Définitions : cluster, noeud, sharding
Nature distribuée d'elasticsearch
Présentation des fonctionnalités : stockage distribué, calculs distribués avec Elasticsearch, tolérance aux pannes.

CB022

ElasticSearch : infrastructure et administration

Fonctionnement	: Notion de noeud maître, stockage des documents : , shard primaire et réplicat, routage interne des requêtes.
Gestion du cluster	: Outils d'interrogation : /_cluster/health Création d'un index : définition des espaces de stockage (shard), allocation à un noeud Configuration de nouveaux noeuds : tolérance aux pannes matérielles et répartition du stockage
Cas d'une panne	: Fonctionnement en cas de perte d'un noeud : élection d'un nouveau noeud maître si nécessaire, déclaration de nouveaux shards primaires
Mise en oeuvre X-Pack Security	: Présentation des apports de X-Pack security: authentification, gestion des accès aux données (rôles), filtrage par adresse IP ; cryptage des données, contrôle des données; audit d'activité.
Exploitation	: Gestion des logs : ES_HOME/logs Paramétrage de différents niveaux de logs : INFO, DEBUG, TRACE Suivi des performances. Sauvegardes avec l'API snapshot.
Evolutions	: Les différentes versions : fonctionnalités et particularités des versions de 2.0 à 5.0. Nouveautés de la version 6.0.

Hadoop : l'écosystème

CB030

Durée: 1 jour
680 €

18 février
23 avril
28 juin
16 septembre
4 novembre

Public:
Chefs de projets, développeurs, et toute personne souhaitant comprendre les mécanismes Hadoop et le rôle de chaque composant.

Objectifs:
Faire le point sur les différents éléments de l'écosystème Hadoop et leurs rôles respectifs.Comprendre l'architecture des applicatifs hadoop et savoir quels sont les apports et les cas d'usage des solutions hadoop.

Connaissances préalables nécessaires:
Connaissances générales des systèmes d'information.

Programme:

Introduction : Rappels sur NoSQL. Le théorème CAP.Historique du projet hadoop.Les fonctionnalités : stockage, outils d'extraction, de conversion, ETL, analyse, ... Exemples de cas d'utilisation sur des grands projets.Les principaux composants :HDFS pour le stockage et YARN pour les calculs. Les distributions et leurs caractéristiques (HortonWorks, Cloudera, MapR, GreenPlum, Apache, ...)

L'architecture : Terminologie : NameNode, DataNode, ResourceManager.Rôle et interactions des différents composants. Présentation des outils d'infrastructure : ambari, avro, zookeeper;de gestion des données : pig, oozie, tez, falcon, pentaho, sqoop, flume;d'interfaçage avec les applications GIS;de restitution et requêtage : webhdfs, hive, hawq, impala, drill, stinger, tajo, mahout, lucene, elasticSearch, Kibana
Les architectures connexes : spark, cassandra

Exemples interactifs : Démonstrations sur une architecture Hadoop multi-noeuds. Mise à disposition d'un environnement pour des exemples de calcul.Travaux pratiques :Recherches dans des données complexes non structurées.

CB030

Hadoop : l'écosystème

Applications : Cas d'usages de hadoop.Calculs distribués sur des clusters hadoop

Hadoop Hortonworks : administration avec Ambari

CB031

Durée: 3 jours
1890 €

19 au 21 février
24 au 26 avril
17 au 19 septembre
5 au 7 novembre

Public:

Chefs de projet, administrateurs et toute personne souhaitant mettre en oeuvre un système distribué avec Hadoop.Les travaux pratiques sont réalisés sur une distribution Hadoop Hortonworks.

Objectifs:

Connaître les principes du framework Hadoop et savoir l'installer, le configurer et l'administrer avec Ambari (tableaux de bord, supervision, gestion des services, etc ...)

Connaissances préalables nécessaires:

Connaissance des commandes des systèmes unix/linux et des bases TCP/IP

Programme:

Introduction : Les fonctionnalités du framework Hadoop.Les différentes versions.
Distributions : Apache, Cloudera, Hortonworks, EMR, MapR, DSE. Spécificités de chaque distribution.
Architecture et principe de fonctionnement. Terminologie : NameNode, DataNode, ResourceManager, NodeManager.
Rôle des différents composants.Le projet et les modules : Hadoop Common, HDFS, YARN, Spark, MapReduce.Oozie, Pig, Hive, HBase, ...

Les outils Hadoop : Infrastructure/Mise en oeuvre :Avro, Ambari, Zookeeper, Pig, Tez, Oozie, Falcon, Pentaho
Vue d'ensemble. Gestion des données.Exemple de sqoop.Restitution : webhdfs, hive, Hawq, Mahout,ElasticSearch ..
Outils complémentaires:Spark, SparkQL, SparkMLib, Storm, BigTop, Zebra, de développement : Cascading, Scalding, Flink/Pachyderm, d'analyse : RHadoop, Hama, Chukwa, kafka

CB031

Hadoop Hortonworks : administration avec Ambari

Installation et configuration : Trois modes d'installation : local, pseudo-distribué, distribué
Première installation.Mise en oeuvre avec un seul noeud Hadoop.Configuration de l'environnement,étude des fichiers de configuration :
core-site.xml, hdfs-site.xml, mapred-site.xml, yarn-site.xml et capacity-scheduler.xml
Création des users pour les daemons hdfs et yarn,droits d'accès sur les exécutables et répertoires.
Lancement des services.Démarrage des composants : hdfs, hadoop-daemon, yarn-daemon, etc ..
Gestion de la grappe, différentes méthodes :ligne de commandes, API Rest, serveur http intégré, APIS natives
Exemples en ligne de commandes avec hdfs, yarn, mapred.
Présentation des fonctions offertes par le serveur http
Travaux pratiques :Organisation et configuration d'une grappe hadoop

Administration Hadoop : Outils complémentaires à yarn et hdfs : jConsole, jconsole yarn. Exemples sur le suivi de charges, l'analyse des journaux.
Principe de gestion des noeuds, accès JMX.
Travaux pratiques : mise en oeuvre d'un client JMX
Administration HDFS :présentation des outils de stockage des fichiers, fsck, dfsadmin.Mise en oeuvre sur des exemples simples de récupération de fichiers
Gestion centralisée de caches avec Cacheadmin. Déplacement d'un NameNode. Mise en mode maintenance.

Haute disponibilité : Mise en place de la haute disponibilité sur une distribution Ambari.
Travaux pratiques :Passage d'un système HDFS en mode HA

Sécurité : Mécanismes de sécurité et mise en oeuvre pratique :activation de la sécurité avec Kerberos dans core-site.xml, et dans hdfs-site.xml pour les NameNode et DataNode. Sécurisation de yarn avec la mise en oeuvre d'un proxy et d'un Linux Container Executor.
Travaux pratiques :Mise en place de la sécurité Kerberos sur une distribution Ambari. Création des utilisateurs. Travaux sur les droits d'accès et les droits d'exécution. Impact au niveau des files Yarn, Oozie et Tez.

Hadoop Hortonworks : administration avec Ambari

CB031

Exploitation : Installation d'une grappe Hadoop avec Ambari. Tableau de bord. Lancement des services.Principe de la supervision des éléments par le NodeManager.
Monitoring graphique avec Ambari.Présentation de Ganglia,Kibana
Travaux pratiques :Visualisation des alertes en cas d'indisponibilité d'un noeud. Configuration des logs avec log4j.

Hadoop Cloudera : administration

CB032

Durée: 3 jours
1890 €

25 au 27 février
20 au 22 mai
2 au 4 octobre
9 au 11 décembre

Public:
Chefs de projet, administrateurs et toute personne souhaitant mettre en oeuvre un système distribué avec Hadoop.Les travaux pratiques sont réalisés sur une distribution Hadoop Cloudera.

Objectifs:
Connaître les principes du framework Hadoop et savoir l'installer et le configurer. Maitriser la configuration et la gestion des services avec Cloudera Manager.

Connaissances préalables nécessaires:
Connaissance des commandes des systèmes unix/linux.

Programme:

Introduction : Les fonctionnalités du framework Hadoop. Les différentes versions.Distributions : Apache, Cloudera, Hortonworks, EMR, MapR, DSE.
Spécificités de chaque distribution.Les apports de la distribution Cloudera.Architecture et principe de fonctionnement.
Terminologie : NameNode, DataNode, ResourceManager, NodeManager.Rôle des différents composants.Le projet et les modules : Hadoop Common, HDFS, YARN, Spark, MapReduce.Oozie, Pig, Hive, HBase, ...

Les outils Hadoop : Infrastructure/Mise en oeuvre :Avro, Ambari, Zookeeper, Pig, Tez, Oozie, Falcon, Pentaho.Vue d'ensemble.Gestion des données.Exemple de sqoop.
Restitution : webhdfs, hive, Hawq, Mahout,ElasticSearch ..
Outils complémentaires:Spark, SparkQL, SparkMLib, Storm, BigTop, Zebra,de développement : Cascading, Scalding, Flink/Pachyderm, d'analyse : RHadoop, Hama, Chukwa, kafka

Hadoop Cloudera : administration

CB032

Installation et configuration	: Présentation de Cloudera Manager. Trois modes d'installation : local, pseudo-distribué, distribué. Première installation.Mise en oeuvre avec un seul noeud Hadoop.Présentation de Cloudera Manager. Configuration de l'environnement,étude des fichiers de configuration : core-site.xml, hdfs-site.xml, mapred-site.xml, yarn-site.xml et capacity-scheduler.xml Création des users pour les daemons hdfs et yarn,droits d'accès sur les exécutables et répertoires. Lancement des services. Démarrage des composants : hdfs, hadoop-daemon, yarn-daemon, etc .. Gestion de la grappe, différentes méthodes :ligne de commandes, API Rest, serveur http intégré, APIS natives Exemples en ligne de commandes avec hdfs, yarn, mapred. Présentation des fonctions offertes par le serveur http Travaux pratiques :Organisation et configuration d'une grappe hadoop avec Cloudera Manager Traitement de données.Requêtage SQL avec Impala.
Administration Hadoop	: Outils complémentaires à yarn et hdfs : jConsole, jconsole yarn. Exemples sur le suivi de charges, l'analyse des journaux. Principe de gestion des noeuds, accès JMX.Travaux pratiques : mise en oeuvre d'un client JMX Administration HDFS :présentation des outils de stockage des fichiers, fsck, dfsadmin Mise en oeuvre sur des exemples simples de récupération de fichiers.Gestion centralisée de caches avec Cacheadmin
Sécurité	: Mécanismes de sécurité et mise en oeuvre pratique :Activation de la sécurité avec Kerberos dans core-site.xml, et dans hdfs-site.xml pour les NameNode et DataNode. Sécurisation de yarn avec la mise en oeuvre d'un proxy et d'un Linux Container Executor.
Exploitation	: Installation d'une grappe Hadoop. Lancement des services.Principe de la supervision des éléments par le NodeManager. Présentation de Ganglia,Kibana Travaux pratiques :Visualisation des alertes en cas d'indisponibilité d'un noeud. Configuration des logs avec log4j.

Hadoop : développement

CB033

Durée: 3 jours
1890 €

4 au 6 mars
24 au 26 juin
23 au 25 septembre
2 au 4 décembre

Public:

Chefs de projets, développeurs, data-scientists, et toute personne souhaitant comprendre les techniques de développement avec MapReduce dans l'environnement Hadoop.

Objectifs:

Connaître les principes du framework Hadoop et savoir utiliser la technologie MapReduce pour paralléliser des calculs sur des volumes importants de données.

Connaissances préalables nécessaires:

Connaissance d'un langage de programmation objet comme Java.

Programme:

Introduction : Les fonctionnalités du framework Hadoop
Le projet et les modules : Hadoop Common, HDFS, YARN, Spark, MapReduce
Utilisation de yarn pour piloter les jobs mapreduce.

MapReduce : Principe et objectifs du modèle de programmation MapReduce.
Fonctions map() et reduce(). Couples (clés, valeurs).
Implémentation par le framework Hadoop.
Etude de la collection d'exemples.
Travaux pratiques :
Rédaction d'un premier programme et exécution avec Hadoop.

Programmation : Configuration des jobs, notion de configuration.
Les interfaces principales : mapper, reducer,
La chaîne de production : entrées, input splits, mapper, combiner, shuffle/sort, reducer, sortie.
partitioner, outputcollector, codecs, compresseurs..
Format des entrées et sorties d'un job MapReduce : InputFormat et OutputFormat.
Travaux pratiques : type personnalisés : création d'un writable spécifique. Utilisation. Contraintes.

Hadoop : développement

CB033

Outils complémentaires	: Mise en oeuvre du cache distribué. Paramétrage d'un job : ToolRunner, transmission de propriétés. Accès à des systèmes externes : S3, hdfs, har, ... Travaux pratiques : répartition du job sur la ferme au travers de yarn.
Streaming	: Définition du streaming map/reduce. Création d'un job map/reduce en python. Répartition sur la ferme. Avantage et inconvénients. Liaisons avec des systèmes externes. Introduction au pont HadoopR Travaux pratiques : suivi d'un job en streaming.
Pig	: Présentation des pattern et best practices Map/reduce. Introduction à Pig. Caractéristiques du langage : latin. Travaux pratiques : installation/lancement de pig. Ecriture de scripts simples pig. Les fonctions de base. Ajouts de fonctions personnalisées. Les UDF. Mise en oeuvre.
Hive	: Simplification du requêtage. Etude de la syntaxe de base. Travaux pratiques : Création de tables. Ecriture de requêtes. Comparaison pig/hive.
Sécurité en environnement hadoop	: Mécanisme de gestion de l'authentification. Travaux pratiques : configuration des ACLs.

Hadoop : stockage avec HBase

CB034

Durée: 2 jours
1190 €

7 et 8 mars
27 au 28 juin
26 au 27 septembre
5 au 6 décembre

Public:

Chefs de projet, administrateurs et toute personne souhaitant stocker des données avec Hbase.

Objectifs:

Comprendre le fonctionnement de HBase, savoir mettre en place une configuration distribuée.

Connaissances préalables nécessaires:

Connaissance des principes de base Hadoop et des bases de données.

Programme:

Introduction : Rappels rapides sur l'ecosystème Hadoop.
Les fonctionnalités du framework Hadoop
Le projet et les modules : Hadoop Common, HDFS, YARN, Spark, MapReduce
Présentation HBase. Historique. Lien avec HDFS.
Format des données dans HBase
Définitions : table, région, ligne, famille de colonnes,cellules, espace de nommage, ...
Fonctionnalités : failover automatique, sharding,
interface avec des jobs MapReduce.

Architecture : HBase master node, Region Master, liens avec les clients HBase.
Présentation du rôle de Zookeeper.

Installation : Choix des packages.
Installation et configuration dans le fichier conf/hbase-site.xml
Démarrage en mode standalone start-hbase.
Test de connexion avec hbase shell.
Installation en mode distribué.
Travaux pratiques :
Interrogations depuis le serveur http intégré.

CB034

Hadoop : stockage avec HBase

HBase utilisation : shell : Présentation des différentes interfaces disponibles.
Travaux pratiques avec hbase shell.
Commandes de base, syntaxe, variables,
manipulation des données : create, list, put, scan, get
désactiver une table ou l'effacer : disable (enable), drop, ...
Programmation de scripts.
Gestion des tables : principe des filtres.
Mise en oeuvre de filtres de recherche, paramètres des tables.
Présentation des espaces de nommage.

Cluster HBase : Fonctionnement en mode distribué
Première étape : fonctionnement indépendant des démons (HMaster, HRegionServer, Zookeeper)
Passage au mode distribué :
mise en oeuvre avec HDFS dans un environnement distribué.
Travaux pratiques :
sur un exemple de tables réparties : mise en oeuvre des splits.

Programmation : Introduction, les APIs (REST, Avro, Thrift, Java, Ruby, ...)
Utilisation d'un client Java.
Gestion des tables.Lien avec MapReduce.
Principe des accès JMX.
Travaux pratiques :
création d'un client JMX

Hadoop : infrastructure sécurisée

CB035

Durée: 1 jour
680 €

25 janvier — 6 septembre
29 avril — 8 novembre

Public:
Chefs de projet, administrateurs et toute personne souhaitant sécuriser une infrastructure hadoop.

Objectifs:
Comprendre les mécanismes de sécurité hadoop, et savoir les mettre en oeuvre.

Connaissances préalables nécessaires:
Connaissance des principes de base Hadoop et des bases de données.

Programme:

Introduction	: Rappels rapides sur l'ecosystème Hadoop. Le projet et les modules : Hadoop Common, HDFS, YARN, Spark, MapReduce Les risques et points à sécuriser dans un système distribué
Architecture sécurité hadoop	: Sécurisation réseau, système d'exploitation, les rôles hadoop et stratégies
Kerberos	: Principe de fonctionnement. Travaux pratiques: kerberisation d'une grappe hadoop
Sécurité des accès	: Authentification, autorisations, accounting.Travaux pratiques: gestion des autorisations dans HDFS, YARN, HBase, ..Mise en oeuvre des ACLs dans Zookeeper
Apache Sentry	: Présentation du projet, architecture : sentry server, sentry plugin. Gestion de l'authentification et des droits d'accès aux données.Travaux pratiques: intégration avec Hadoop
Sécurité des données	: Cryptage des données stockées, et en transit. Mécanisme de sécurité des données en entrée et en consultation par des accès clients: interface hadoop en ligne de commande, sqoop, oozie, HBase, webHDFS, httpFS

Hadoop : analytics

CB036

Durée: 2 jours
1190 €

7 au 8 février — 19 au 20 septembre
16 au 17 mai — 19 au 20 décembre

Public:
Chefs de projet, développeurs, data scientists, architectes souhaitant mettre en oeuvre des solutions analytics avec hadoop.

Objectifs:
Savoir mettre en oeuvre les frameworks analytics dans un environnement hadoop.

Connaissances préalables nécessaires:
Connaissances des principes du BigData, d'un langage de programmation comme Java ou Scala ou Python.

Programme:

Introduction : Définitions : Analytics.
Arbres de décision, de régression, régression automatique
Classifieurs, scoring.
Apprentissage supervisé, apprentissage automatique
Etapes de préparation des données.
Présentation du data munging.

Hadoop et les outils d'analyse : Rôle des différents composants :
socle hadoop, yarn, hdfs
Frameworks analytics : Mahout, Flink, Spark ML

Mahout : Principe de fonctionnement.
Sources de données, format de stockage des données,
Génération de recommandations, traitement, filtrage
Exemples de base : génération de recommandations, traitement, filtrage
Présentation des algorithmes les plus courants.
Compatibilité avec Hadoop Yarn, Spark, H2O, Flink

CB036

Hadoop : analytics

Flink : Origine du projet, fonctionnalités
Traitement distribué de flux de données, en temps réel ou batch
APIs disponibles
Mise en oeuvre avec des programmes Java/Scala
Analyse de graphe avec l'API Gelly

Spark MLib : Fonctionnalités : Machine Learning avec Spark,
algorithmes standards,
gestion de la persistence,statistiques.
Support de RDD.
Mise en oeuvre avec les DataFrames.

GraphX : Fourniture d'algorithmes, d'opérateurs simples pour des calculs statistiques sur les graphes
Travaux pratiques :
exemples d'opérations sur les graphes.

Spark : traitement de données

CB037

Durée: 3 jours
1800 €

25 au 27 mars — 21 au 23 octobre
17 au 19 juin — 16 au 18 décembre

Public:
Chefs de projet, data scientists, développeurs.

Objectifs:
Comprendre le fonctionnement de Spark et son utilisation dans un environnement Hadoop. Savoir intégrer Spark dans un environnement Hadoop, traiter des données Cassandra, HBase, Kafka, Flume, Sqoop, S3. Ce stage permet de se présenter à l'examen "Certification Hadoop avec Spark pour développeur de Cloudera"

Connaissances préalables nécessaires:
Connaissance de Java ou Python, notions de calculs statistiques et des bases Hadoop ou avoir suivi le stage "Hadoop, l'écosystème".

Programme:

Introduction : Présentation Spark, origine du projet,apports, principe de fonctionnement.Langages supportés.

Premiers pas : Utilisation du shell Spark avec Scala ou Python.Modes de fonctionnement. Interprété, compilé.
Utilisation des outils de construction. Gestion des versions de bibliothèques.

Règles de développement : Mise en pratique en Java, Scala et Python.Notion de contexte Spark
Différentes méthodes de création des RDD : depuis un fichier texte, un stockage externe.
Manipulations sur les RDD (Resilient Distributed Dataset).Fonctions, gestion de la persistence.

CB037

Spark : traitement de données

Cluster	: Différents cluster managers : Spark en autonome, avec Mesos, avec Yarn, avec Amazon EC2 Architecture : SparkContext, Cluster Manager, Executor sur chaque noeud.Définitions : Driver program, Cluster manager, deploy mode, Executor, Task, Job Mise en oeuvre avec Spark et Amazon EC2.Soumission de jobs, supervision depuis l'interface web
Traitements	: Lecture/écriture de données : Texte, JSon, Parquet, HDFS, fichiers séquentiels. Jointures. Filtrage de données, enrichissement.Calculs distribués de base. Introduction aux traitements de données avec map/reduce. Travail sur les RDDs. Transformations et actions. Lazy execution. Impact du shuffle sur les performances. RDD de base, key-pair RDDs.Variables partagées : accumulateurs et variables broadcast.
Intégration hadoop	: Présentation de l'écosystème Hadoop de base : HDFS/Yarn.Travaux pratiques avec YARN Création et exploitation d'un cluster Spark/YARN.Intégration de données sqoop, kafka, flume vers une architecture Hadoop. Intégration de données AWS S3.
Support Cassandra	: Description rapide de l'architecture Cassandra. Mise en oeuvre depuis Spark.Exécution de travaux Spark s'appuyant sur une grappe Cassandra.
DataFrames	: Spark et SQL Objectifs : traitement de données structurées,.L'API Dataset et DataFrames Optimisation des requêtes.Mise en oeuvre des Dataframes et DataSet.Comptabilité Hive Travaux pratiques: extraction, modification de données dans une base distribuée.Collections de données distribuées.Exemples.
Streaming	: Objectifs , principe de fonctionnement : stream processing.Source de données : HDFS, Flume, Kafka, ... Notion de StreamingContexte, DStreams, démonstrations.Travaux pratiques : traitement de flux DStreams en Scala.

Spark : traitement de données

CB037

Machine Learning: Fonctionnalités : Machine Learning avec Spark,algorithmes standards, gestion de la persistence, statistiques.
Support de RDD.Mise en oeuvre avec les DataFrames.

Spark GraphX : Fourniture d'algorithmes, d'opérateurs simples pour des calculs statistiques sur les graphes
Travaux pratiques :exemples d'opérations sur les graphes.

Flux de données avec Storm

CB038

Durée: 2 jours
1190 €

28 au 29 mars
20 au 21 juin
24 au 25 octobre
19 au 20 décembre

Public:
Chefs de projet, data scientists, développeurs.

Objectifs:
Savoir mettre en oeuvre Storm pour le traitement de flux de données.

Connaissances préalables nécessaires:
Connaissance d'un langage de programmation comme Java ou Python.

Programme:

Introduction : Présentation de Storm:fonctionnalités, architecture, langages supportés
Définitions:spout, bolt, topology

Architecture : Etude des composants d'un cluster Storm : master node 'nimbus' et worker nodes
Positionnement par rapport à un cluster Hadoop.Le modèle de données. Différents types de flux.

Premiers pas : Configuration d'un environnement de développement.Installation d'un cluster Storm. Travaux pratiques sur le projet storm-starter

Flux de données : Définition du nombre de flux dans un noeud, création de topologies regroupants des flux entre différents noeuds,
communication entre flux en JSON, lecture de flux d'origines diverses (JMS, Kafka, ...)

Haute disponibilité : Tolérance aux pannes: principe de fiabilisation des master node, workers node, nimbus
Garantie de traitement des flux: principe,paramètres TOPOLOGY_MESSAGE_TIMEOUT_SECS, TOPOLOGY_ACKERS
Traitements temps réel avec Trident. Scalabilité: parallèlisme dans un cluster storm, ajouts de noeuds, commande 'storm rebalance'

CB039

Programmation Scala

Durée: 3 jours
1800 €

4 au 6 février — 2 au 4 septembre
13 au 15 mai — 9 au 11 décembre

Public:
Chefs de projet, data scientists, développeurs.

Objectifs:
Comprendre les apports du langage Scala, de la programmation fonctionnelle. Maîtriser la programmation Scala, savoir s'interfacer avec des programmes Java.

Connaissances préalables nécessaires:
Connaissance d'un langage de programmation et de la programmation objet.

Programme:

Introduction : Présentation de Scala. Les points forts du langage : extensibilité, programmation objet, programmation fonctionnelle, utilisation de la JVM.

Premiers pas : Différents modes d'utilisation de Scala : compilé, en script, avec un interpréteur
Outils de développement Scala : compilateur scalac, sbt (Scala's Build Tool), IntelliJ avec le plugin Scala
Travaux pratiques:réalisation de programmes simples (calcul et affichage)

Syntaxe : Les variables, les fonctions, les classes, les traits.Le cas particulier des "singleton objects" et "companion objets"
Les opérateurs. Les annotations.

Programmation fonctionnelle : Principe et différences par rapport à la programmation impérative
Particularités sur les tuples, listes, tables associatives.

Interfaçage avec Java : Fonctionnement de scala, byte code.Différences entre Java et Scala.Appel de classes Scala depuis du code Java
Utilisation de bibliothèques Java dans un programme Scala

Pig : développement de scripts

CB040

Durée: 2 jours
1360 €

7 au 8 février
16 au 17 mai
5 au 6 septembre
11 au 12 décembre

Public:

Chefs de projet, data scientists, développeurs souhaitant utiliser pig pour l'analyse de données

Objectifs:

Comprendre le fonctionnement de pig, savoir développer des requêtes en latin, pour effectuer des transformations sur des données,des analyses de données,intégrer des données de différents formats.

Connaissances préalables nécessaires:

Connaissance de Java ou Python, des bases Hadoop, et notions de calculs statistiques.

Programme:

Introduction : Le projet Apache Pig, fonctionnalités, versions
Présentation de Pig dans l'écosystème Hadoop.
Chaîne de fonctionnement.
Comparatif avec l'approche Hive ou Spark

Mise en oeuvre : Rappels sur les commandes HDFS
Prérequis techniques, configuration de Pig
Travaux pratiques:
Exécution : les différents modes : interactif ou batch
Principe de l'exécution de scripts Pig Latin avec Grunt

Base latin : Modèles de données avec Pig
Intégration Pig avec MapReduce
Les requêtes Latin : chargement de données, instructions
Ordres de bases :
LOAD, FOREACH, FILTER, STORE.
Travaux pratiques : création d'un ETL de base
Contrôle d'exécution

Transformations : Groupements, jointures, tris, produits cartésiens.
Transformation de base de la donnée.
Découpages. Découpages sur filtres.

Pig : développement de scripts

CB040

Analyse de la donnée	: Echantillonages. Filtres. Rangements avec rank et dense. Calculs : min/max, sommes, moyennes, ... Ttravaux pratiques : Traitements de chaînes de caractères. Traitement de dates.
Intégration	: Formats d'entrées/sorties. Interfaçage avro, json. Travaux pratiques : chargement de données depuis HDFS vers HBase, analyse de données Pig/Hbase et restitution Json.
Extensions	: Extension du PigLatin. Création de fonctions UDF en java. Intégration dans les scripts Pig. Travaux pratiques : Utilisation de Pig Latin depuis des programmes Python Execution de programmes externes, streaming.

Kafka, ingestion et traitement de messages

CB041

Durée: 1 jour
660 €

1er mars — 16 septembre
24 mai — 7 novembre

Public:
Chefs de projet, développeurs souhaitant mettre enoeuvre kafka pour la distribution de mesages.

Objectifs:
Comprendre le fonctionnement de kafka, acquérir les bonnes pratiques de distribution de messages, savoir configurer kafka pour intégrer les données de différents formats et de sources différentes.

Connaissances préalables nécessaires:
Connaissance de l'écosystème hadoop et bases de programmation dans un langage objet (java ou scala ou python)

Programme:

Introduction : Le projet Kafka : historique, fonctionnalités, principe de fonctionnement.
Présentation de l'achitecture et du rôle de chaque composant :
broker, producer, consumer
Liaison avec Zookeeper

Mise en oeuvre : Préconisations d'installation et prérequis
Travaux pratiques:
installation et lancement de zookeeper et du kafka-server,
Création d'un topic simple,
Mise en oeuvre d'une chaîne de base.
Visualisation des messages avec kafka-console-consumer

Multi-broker : Etude de la configuration du broker
Travaux pratiques :
création d'une configuration multi-broker,
démarrage de plusieurs noeuds

CB041

Kafka, ingestion et traitement de messages

La réplication : Facteur de réplication
Partitions
Travaux pratiques:
tests de haute disponibilité dans une configuration multi-noeuds

Kafka Connect : Présentation des fonctionnalités : intégration de données d'origines multiples,
modes de fonctionnement (standalone ou distribué)
Travaux pratiques :
configuration de connecteurs, ingestion de données,
création d'une chaîne de transformation

Kafka Streams : Les apports de Kafka Streams: applications temps réel et microservices

Data Classification et Machine Learning

CB050

Durée: 2 jours
1330 €

11 et 12 février
20 et 21 mai
9 et 10 septembre
4 au 5 novembre

Public:

Chefs de projet, développeurs, data scientists, architectes souhaitant comprendre comment organiser le traitement des données et structurer les processus de Machine Learning.

Objectifs:

Savoir définir les étapes de préparation des données, comprendre et mettre en oeuvre l'apprentissage automatique, les techniques de classification de données, les apports des réseaux de neurones et du Deep Learning.

Connaissances préalables nécessaires:

Connaissances des principes du BigData, et des architectures techniques mises en oeuvre.

Programme:

Introduction : Zoom sur les données : format, volumes, structures, ... et les requêtes, attentes des utilisateurs.
Etapes de la préparation des données.
Définitions, présentation du data munging
Le rôle du data scientist.

Gouvernance des données : Qualité des données.
Transformation de l'information en donnée. Qualification et enrichissement.
Sécurisation et étanchéité des lacs de données.
Flux de données et organisation dans l'entreprise. De la donnée maître à la donnée de travail. MDM.
Mise en oeuvre pratique des différentes phases : nettoyage,enrichissement,organisation des données.

Traitements statistiques de base : Introduction aux calculs statistiques. Paramétrisation des fonctions.
Applications aux fermes de calculs distribués. Problématiques induites. Approximations. Précision des estimations.

CB050

Data Classification et Machine Learning

Data Mining	: Besoin, apports et enjeux. Extraction et organisation des classes de données. Analyse factorielle.
Machine Learning	: Apprentissage automatique Définition, les attentes par rapport au Machine Learning Les valeurs d'observation, et les variables cibles. Ingénierie des variables. Les méthodes : apprentissage supervisé et non supervisé Classification des données, Algorithmes : régression linéaire, k-moyennes, k-voisins, classification naïve bayésienne, arbres de décision, forêts aléatoires, etc .. Création de jeux d'essai, entraînement et construction de modèles. Prévisions à partir de données réelles. Mesure de l'efficacité des algorithmes. Courbes ROC. Parallélisation des algorithmes. Choix automatique.
IA	: Introduction aux réseaux de neurones. Réseaux de neurones à convolution. Modèles de CNN. Les types de couches : convolution, pooling et pertes. L'approche du Deep Learning. Deeplearning4j sur Spark.
Les risques et écueils	: Importance de la préparation des données. L'écueil du "surapprentissage".
Visualisation des données	: L'intérêt de la visualisation. Outils disponibles, Exemples de visualisation avec R et Python

Dataviz : solutions opensource

CB051

Durée: 2 jours
1360 €

7 au 8 février
16 au 17 mai
19 au 20 septembre
19 au 20 décembre

Public:
Chefs de projet, architectes, développeurs, data-scientists souhaitant mettre en oeuvre des solutions d'analyse et de visualisation des données

Objectifs:
Savoir quelles sont les solutions et outils opensource d'analyse et de restitution de données, connaître les spécificités, contraintes et apports des différentes solutions.

Connaissances préalables nécessaires:
Connaissances générales sur le Bigdata, le data-mining, l'analyse de données.

Programme:

Introduction : Fonctionnalités des outils de dataviz : analyses statistiques, classifications, rapprochements,production de recommandations, représentations graphiques,
Présentation de quelques outils : Mahout, Giraph, Agile, spagobi, Neo4j, Spark GraphX, Gephi
Positionnement et fonctionnalités.

Mahout : Présentation Mahout.
Positionnement dans l'offre BigData et Machine Learning : Hadoop, Spark,..
Fonctionnalités. Mode autonome ou mode distribué
Exemples d'algorithmes fournis avec Mahout.

Giraph : Principe du projet Giraph. Algorithmes de graphes.Infrastructure Hadoop.Cas d'utilisation.

spagoBI : Historique de spagoBI, positionnement, fonctionnalités.
Installation et démonstrations depuis l'interface web.
Exemples d'analyse avec Cockpit.

Dataviz : solutions opensource

CB051

Neo4j : Présentation du produit. Requêtage avec le Cypher Query Language
Travaux pratiques : création d'un graphe,
Requêtes de recherche, navigation dans le graphe

Spark GraphX : Fourniture d'algorithmes, d'opérateurs simples pour des calculs statistiques sur les graphes
Travaux pratiques :
exemples d'opérations sur les graphes.

Gephi : Mise en oeuvre. Gestion des graphes. Positionnement des objets.
Import de données.
Travaux pratiques :
Mise en oeuvre avec Spark/GraphX

Environnement R, analyse de données

CB052

Durée: 3 jours
2040 €

13 au 15 février
22 au 24 mai
11 au 13 septembre
6 au 8 novembre

Public:

Chefs de projet, data scientists, statisticiens, développeurs souhaitant comprendre les apports de R pour l'analyse des données, et savoir l'intégrer à un environnement Hadoop.

Objectifs:

Connaître les principales fonctions statistiques de R, et savoir utiliser des programmes R dans un environnement BigData, en s'appuyant sur le système distribué hdfs.

Connaissances préalables nécessaires:

Notions de calculs statistiques

Programme:

Présentation R : Le projet R Programming
Calculs statistiques et génération de graphiques
Points forts de R Programming
Besoins du BigData
Positionnement R programming par rapport à Hadoop

Mise en oeuvre de R : Travaux pratiques : installation et tests sur une plate-forme CentOS
Utilisation de R en mode commande.
Commandes de base. Syntaxe.
Opérations de base. Expressions.
Manipulations de nombres, vecteurs, tableaux, matrices.listes, etc ..

Tableaux et matrices : Déclaration, dimensionnement, indexation.
Opérations de base : produit de tableaux, transposition, produits de matrices.
Matrices : équations linéaires, inversion, valeur propre, vecteur propre, déterminant, moindre carré, ...

Environnement R, analyse de données

CB052

Liste et DataFrames	: Définitions, cas d'utilisation. Attachement, détachement. Chargement d'un dataframe. La fonction scan.
Statistiques	: Distributions embarquées : uniforme, normale, poisson, exponentielle, ... Calculs statistiques. Modèles statistiques. Affichage en graphes, histogrammes.
Import/export	: Formats texte, csv, xml, binaire, largeur fixe, images (jpeg, png). Encodage. Filtrage. Importation SQL. Importation depuis un socket réseau. Travaux pratiques : importation de données géodésiques et export au format Json
Intégration Hadoop	: Association de la puissance du calcul distribué fourni par les outils hadoop Différents moyens d'intégration : sparkR, RHbase, RHDFS, RHadoop, rmr2 pour utiliser le système distribué hdfs depuis R, pour accéder à HBase depuis les programmes en R. Transformation d'un dataframe R en un dataframe Spark. Travaux pratiques avec Hadoop
Fonctions spécifiques	: Définition de nouvelles fonctions. Appels. Passage d'argument. Construction d'une bibliothèque. Diffusion, installation avec R CMD INSTALL.
Evolutions	: Les acteurs : IBM avec BigInsights, Revolution R avec ScaleR

Talend Open Studio, intégration de données

CB060

Durée: 3 jours
1980 €

18 au 20 février
15 au 17 avril
2 au 4 septembre
14 au 16 octobre

Public:

Développeurs, chefs de projet et toute personne souhaitant utiliser Talend OpenStudio pour le traitement de données

Objectifs:

Savoir créer des jobs dans l'application ETL Talend, et les optimiser par l'utilisation des contextes. Savoir exécuter les jobs et en suivre l'exécution par l'utilisation de statistiques.

Connaissances préalables nécessaires:

Maîtrise des SGBDR et de SQL. La connaissance de la programmation en Java serait utile.

Programme:

Introduction	: Rappels sur les solutions ETL. Présentation Talend OpenStudio : installation, configuration des préférences utilisateurs. Documentation. Concevoir des jobs simples avec Talend OpenStudio.
Modélisation	: Présentation des outils : Business Modeler, JobDesigner Mise en oeuvre des principales connexions. Intégration de fichiers XML et CSV Etude des composants de transformation. Analyse du code et exécution des jobs.
Optimisation des jobs	: Utilisation des métadonnées, import/export, propagation sur les jobs, configuration de connexions réutilisables Stockage des variables de contexte dans les fichiers .properties et .ini
Liens avec les bases de données	: Présentation des bases de données supportées Opérations sur les tables, Connexion à un schéma de bases de données Gestion des transactions Utilisation de SQLBuilder pour créer des requêtes

Talend Open Studio, intégration de données

CB060

Traitement de données multi-sources	: Le composant tMap Mise en oeuvre : Création de jointures, transformations à l'aide des variables, expressions et jointures, qualification des données à l'aide de filtres, Génération de sorties multiples Extensions : décomposition de jobs, mise en oeuvre du tRunJob, debugging, analyse des statistiques d'exécution, reporting avec jJasperOutput

Talend Open Studio, optimisation flux de données

CB061

Durée: 2 jours
1520 €

21 au 22 février
18 au 19 avril
5 au 6 septembre
17 au 18 octobre

Public:

Développeurs, chefs de projet et toute personne souhaitant utiliser Talend OpenStudio pour optimiser les flux de données.

Objectifs:

Savoir optimiser les flux de données dans les traitements avec Talend OpenStudio, intégrer du code Java, personnaliser des composants, et optimiser les performances des jobs.

Connaissances préalables nécessaires:

Connaissance de base de Talend OpenStudio, ou avoir suivi le stage "Talend OpenStudio, intégration de données". Maîtrise des SGBDR et de SQL. La connaissance de la programmation en Java serait utile.

Programme:

Rappels Talend OpenStudio	: Rappels rapides sur les ETLs. Fonctionnalités de Talend OpenStudio.
Utilisation avancée	: Gestion des contextes.Les transformations complexes.Principes des composants ELT, et mise en oeuvre Utilisation des expressions régulières
Intégration de composants Java	: Fonctionnement des composants Java.Analyse d'exemples de composants Talend Java. Travaux pratiques : création et modiication de composants,exportation dans la palette. Méthodes d'internationalisation des composants
Performances	: Supervision et journalisation des jobs.Points d'optimisation. Travaux pratiques :modification des variables en mémoire
Production	: Exécution de jobs en dehors de Talend Bonnes pratiques de déploiement de jobs Talend en production

CB080

Cycle Certifiant Data scientist

Durée: 11 jours

Prix et dates: nous consulter

Public:
Chefs de projet, data scientists, statisticiens, développeurs.

Objectifs:
Comprendre les concepts et les apports des technologies BigData, savoir définir les étapes de préparation des données, et les algorithmes de Machine Learning. Connaître les apports des outils comme Hadoop, Spark, le rôle des différents composants, la mise en oeuvre pour du stream processing, des traitements de Machine Learning, des calculs statistiques avec les graphes.

Connaissances préalables nécessaires:
Connaissance des bases des systèmes d'information, et notions de calculs statistiques.

Programme:
Introduction : L'essentiel du BigData : calcul distribué, données non structurées.
Besoins fonctionnels et caractéristiques techniques des projets.La valorisation des données.Le positionnement respectif des technologies de cloud, BigData et noSQL, et les liens, implications.
Concepts clés : ETL, Extract Transform Load, CAP, 3V, 4V, données non structurées, prédictif, Machine Learning.
Quelques applications : Watson (IBM), Amazon Rekognition.L'écosystème du BigData : les acteurs, les produits, état de l'art.
Cycle de vie des projets BigData. Emergence de nouveaux métiers : Datascientists, Data labs, Hadoop scientists, CDO, ...Rôle de la DSI dans la démarche BigData.Gouvernance des données :
importance de la qualité des données, fiabilité, durée de validité, sécurité des données
Aspects législatifs : sur le stockage, la conservation de données, etc ...sur les traitements, la commercialisation des données, des résultats

CB080

Cycle Certifiant Data scientist

Stockage distribué	: Caractéristiques NoSQL. Les différents modes et formats de stockage. Les types de bases de données : clé/valeur, document, colonne, graphe. Besoin de distribution. Définition de la notion d'élasticité. Principe du stockage réparti : Définitions : réplication, sharding, gossip protocol, hachage, Systèmes de fichiers distribués : GFS, HDFS, Ceph. Les bases de données : Cassandra, DynamoDB, Accumulo, HBase, MongoDB, CouchBase, Riak, BigTable, .. Caractéristiques NoSQL :structure de données proches des utilisateurs, développeurs Les types de bases de données : clé/valeur, document, colonne, graphe. Données structurées et non structurées, documents, images, fichiers XML, JSON, CSV, ... Les différents modes et formats de stockage.Stockage réparti : réplication, sharding, gossip protocol, hachage, Systèmes de fichiers distribués : GFS, HDFS,Quelques exemples de produits et leurs caractéristiques : Cassandra, MongoDB, CouchDB, DynamoDB, Riak, Hadoop, HBase, BigTable, ... Qualité des données, gouvernance de données.
Indexation et recherche	: Moteurs de recherche.Principe de fonctionnement.Méthodes d'indexation. Mise en oeuvre avec elasticsearch. Exemple de Lucene/solr.Recherche dans les bases de volumes importants.Exemples de produits et comparaison : Dremel, Drill, ElasticSearch, MapReduce,
Calcul et restitution, intégration	: Différentes solutions : calculs en mode batch, ou en temps réel,sur des flux de données ou des données statiques. Les produits : langage de calculs statistiques, R Statistics Language, sas, RStudio;outils de visualisation : Tableau, QlikView Ponts entre les outils statistiques et les bases BigData Outils de calcul sur des volumes importants : storm en temps réel, hadoop en mode batch. Zoom sur Hadoop : complémentarité de HDFS et MapReduce. Restitution et analyse : logstash, kibana, elk, pentaho Présentation de pig pour la conception de tâches MapReduce sur une grappe Hadoop.

Cycle Certifiant Data scientist

CB080

Présentation hadoop	: Historique du projet hadoop Les fonctionnalités : stockage, outils d'extraction, de conversion, ETL, analyse, ... Exemples de cas d'utilisation sur des grands projets. Les principaux composants :HDFS pour le stockage et YARN pour les calculs. Les distributions et leurs caractéristiques (HortonWorks, Cloudera, MapR, GreenPlum, Apache, ...)
L'architecture	: Terminologie : NameNode, DataNode, ResourceManager.Rôle et interactions des différents composants Présentation des outils d'infrastructure : ambari, avro, zookeeper;de gestion des données : pig, oozie, tez, falcon, pentaho, sqoop, flume; d'interfaçage avec les applications GIS;de restitution et requêtage : webhdfs, hive, hawq, impala, drill, stinger, tajo, mahout, lucene, elasticSearch, Kibana Les architectures connexes : spark, cassandra
Exemples interactifs	: Démonstrations sur une architecture Hadoop multi-noeuds. Mise à disposition d'un environnement pour des exemples de calcul Travaux pratiques :Recherches dans des données complexes non structurées.
Applications	: Cas d'usages de hadoop.Calculs distribués sur des clusters hadoop
Présentation Spark	: Présentation Spark, origine du projet,apports, principe de fonctionnement.Langages supportés.
Premiers pas	: Utilisation du shell Spark avec Scala ou Python Modes de fonctionnement. Interprété, compilé. Utilisation des outils de construction. Gestion des versions de bibliothèques.
Règles de développement	: Mise en pratique en Java, Scala et Python. Notion de contexte Spark Différentes méthodes de création des RDD : depuis un fichier texte, un stockage externe. Manipulations sur les RDD (Resilient Distributed Dataset). Fonctions, gestion de la persistence.

Cycle Certifiant Data scientist

CB080

Cluster	: Différents cluster managers : Spark en autonome, avec Mesos, avec Yarn, avec Amazon EC2 Architecture : SparkContext, Cluster Manager, Executor sur chaque noeud. Définitions : Driver program, Cluster manager, deploy mode, Executor, Task, Job Mise en oeuvre avec Spark et Amazon EC2.Soumission de jobs, supervision depuis l'interface web
Traitements	: Lecture/écriture de données : Texte, JSon, Parquet, HDFS, fichiers séquentiels. Jointures. Filtrage de données, enrichissement.Calculs distribués de base. Introduction aux traitements de données avec map/reduce. Travail sur les RDDs. Transformations et actions. Lazy execution. Impact du shuffle sur les performances. RDD de base, key-pair RDDs.Variables partagées : accumulateurs et variables broadcast.
Intégration hadoop	: Présentation de l'écosystème Hadoop de base : HDFS/Yarn Travaux pratiques avec YARN. Création et exploitation d'un cluster Spark/YARN.Intégration de données sqoop, kafka, flume vers une architecture Hadoop. Intégration de données AWS S3.
Support Cassandra	: Description rapide de l'architecture Cassandra. Mise en oeuvre depuis Spark.Exécution de travaux Spark s'appuyant sur une grappe Cassandra.
DataFrames	: Spark et SQL.Objectifs : traitement de données structurées,L'API Dataset et DataFrames Optimisation des requêtes.Mise en oeuvre des Dataframes et DataSet.Comptabilité Hive Travaux pratiques: extraction, modification de données dans une base distribuée. Collections de données distribuées. Exemples.
Streaming	: Objectifs , principe de fonctionnement : stream processing. Source de données : HDFS, Flume, Kafka, ... Notion de StreamingContexte, DStreams, démonstrations Travaux pratiques : traitement de flux DStreams en Scala.

CB080

Cycle Certifiant Data scientist

Machine Learning	: Fonctionnalités : Machine Learning avec Spark,algorithmes standards, gestion de la persistence, statistiques. Support de RDD.Mise en oeuvre avec les DataFrames.
Spark GraphX	: Fourniture d'algorithmes, d'opérateurs simples pour des calculs statistiques sur les graphes Travaux pratiques :exemples d'opérations sur les graphes.
Data Classification et Machine Learning	: Zoom sur les données : format, volumes, structures, ...et les requêtes, attentes des utilisateurs.Etapes de la préparation des données. Définitions, présentation du data mungingLe rôle du data scientist.
Gouvernance des données	: Qualité des données.Transformation de l'information en donnée. Qualification et enrichissement.Sécurisation et étanchéité des lacs de données. Flux de données et organisation dans l'entreprise. De la donnée maître à la donnée de travail. MDM. Mise en oeuvre pratique des différentes phases :nettoyage,enrichissement,organisation des données.
Traitements statistiques de base	: Introduction aux calculs statistiques. Paramétrisation des fonctions.Applications aux fermes de calculs distribués. Problématiques induites. Approximations. Précision des estimations.
Data Mining	: Besoin, apports et enjeux.Extraction et organisation des classes de données.Analyse factorielle.
Machine Learning	: Apprentissage automatique. Définition, les attentes par rapport au Machine Learning Les valeurs d'observation, et les variables cibles.Ingénierie des variables. Les méthodes : apprentissage supervisé et non supervisé. Classification des données, Algorithmes : régression linéaire, k-moyennes, k-voisins, classification naïve bayésienne, arbres de décision, forêts aléatoires, etc .. Création de jeux d'essai, entraînement et construction de modèles.Prévisions à partir de données réelles. Mesure de l'efficacité des algorithmes. Courbes ROC. Parallélisation des algorithmes. Choix automatique.

Cycle Certifiant Data scientist

CB080

IA	: Introduction aux réseaux de neurones.Réseaux de neurones à convolution. Modèles de CNN. Les types de couches : convolution, pooling et pertes.L'approche du Deep Learning. Deeplearning4j sur Spark.
Les risques et écueils	: Importance de la préparation des données.L'écueil du "surapprentissage".
Visualisation des données	: L'intérêt de la visualisation.Outils disponibles, Exemples de visualisation avec R et Python
Apache Pig	: Le projet Apache Pig, fonctionnalités, versions.Présentation de Pig dans l'écosystème Hadoop. Chaîne de fonctionnement. Comparatif avec l'approche Hive ou Spark
Mise en oeuvre	: Rappels sur les commandes HDFS. Prérequis techniques, configuration de Pig. Travaux pratiques: Exécution : les différents modes : interactif ou batch Principe de l'exécution de scripts Pig Latin avec Grunt
Base latin	: Modèles de données avec Pig.Intégration Pig avec MapReduce. Les requêtes Latin : chargement de données, instructions Ordres de bases :LOAD, FOREACH, FILTER, STORE. Travaux pratiques : création d'un ETL de base. Contrôle d'exécution
Transformations	: Groupements, jointures, tris, produits cartésiens.Transformation de base de la donnée.Découpages. Découpages sur filtres.
Analyse de la donnée	: Echantillonages. Filtres. Rangements avec rank et dense.Calculs : min/max, sommes, moyennes, ... Travaux pratiques :Traitements de chaînes de caractères. Traitement de dates.

Cycle Certifiant Data scientist

CB080

Intégration	: Formats d'entrées/sorties. Interfaçage avro, json. Travaux pratiques : chargement de données depuis HDFS vers HBase, analyse de données Pig/Hbase et restitution Json.
Extensions	: Extension du PigLatin.Création de fonctions UDF en java.Intégration dans les scripts Pig. Travaux pratiques :Utilisation de Pig Latin depuis des programmes Python Execution de programmes externes, streaming.
Kafka	: Le projet Kafka : historique, fonctionnalités, principe de fonctionnement. Présentation de l'achitecture et du rôle de chaque composant :broker, producer, consumer Liaison avec Zookeeper
Mise en oeuvre	: Préconisations d'installation et prérequis Travaux pratiques:installation et lancement de zookeeper et du kafka-server, Création d'un topic simple,Mise en oeuvre d'une chaîne de base. Visualisation des messages avec kafka-console-consumer
Multi-broker	: Etude de la configuration du broker Travaux pratiques :création d'une configuration multi-broker,démarrage de plusieurs noeuds
La réplication	: Facteur de réplication. Partitions. Travaux pratiques:tests de haute disponibilité dans une configuration multi-noeuds
Kafka Connect	: Présentation des fonctionnalités : intégration de données d'origines multiples, modes de fonctionnement (standalone ou distribué) Travaux pratiques : configuration de connecteurs, ingestion de données, création d'une chaîne de transformation
Kafka Streams	: Les apports de Kafka Streams: applications temps réel et microservices

Cycle Certifiant architecte BigData

CB090

Durée: 10 jours

Prix et dates: nous consulter

Public:
Chefs de projet,architectes

Objectifs:
Comprendre les concepts et les apports des technologies BigData. Connaître les apports des outils comme Hadoop, Spark, le rôle des différents composants, la mise en oeuvre pour du stream processing, des traitements de Machine Learning, des calculs statistiques avec les graphes.

Connaissances préalables nécessaires:
Connaissance des bases des systèmes d'information, et notions de calculs statistiques.

Programme:
Introduction : L'essentiel du BigData : calcul distribué, données non structurées.Besoins fonctionnels et caractéristiques techniques des projets.La valorisation des données.Le positionnement respectif des technologies de cloud, BigData et noSQL, et les liens, implications.
Quelques éléments d'architecture.Concepts clés : ETL, Extract Transform Load, CAP, 3V, 4V, données non structurées, prédictif, Machine Learning.
Quelques applications : Watson (IBM), Amazon Rekognition
L'écosystème du BigData : les acteurs, les produits, état de l'art.
Cycle de vie des projets BigData. Emergence de nouveaux métiers : Datascientists, Data labs, Hadoop scientists, CDO, ...
Rôle de la DSI dans la démarche BigData. Gouvernance des données :importance de la qualité des données, fiabilité, durée de validité, sécurité des données
Aspects législatifs : sur le stockage, la conservation de données, etc ...sur les traitements, la commercialisation des données, des résultats

Cycle Certifiant architecte BigData

CB090

Stockage distribué	: Caractéristiques NoSQL. Les différents modes et formats de stockage. Les types de bases de données : clé/valeur, document, colonne, graphe. Besoin de distribution. Définition de la notion d'élasticité.Principe du stockage réparti : Définitions : réplication, sharding, gossip protocol, hachage, Systèmes de fichiers distribués : GFS, HDFS, Ceph Les bases de données : Cassandra, DynamoDB, Accumulo, HBase, MongoDB, CouchBase, Riak, BigTable, .. Caractéristiques NoSQL :structure de données proches des utilisateurs, développeurs Les types de bases de données : clé/valeur, document, colonne, graphe. Données structurées et non structurées, documents, images, fichiers XML, JSON, CSV, ... Les différents modes et formats de stockage.Stockage réparti : réplication, sharding, gossip protocol, hachage, Systèmes de fichiers distribués : GFS, HDFS,Quelques exemples de produits et leurs caractéristiques : Cassandra, MongoDB, CouchDB, DynamoDB, Riak, Hadoop, HBase, BigTable, ... Qualité des données, gouvernance de données.
Indexation et recherche	: Moteurs de recherche.Principe de fonctionnement. Méthodes d'indexation. Mise en oeuvre avec elasticsearch. Exemple de Lucene/solr. Recherche dans les bases de volumes importants. Exemples de produits et comparaison : Dremel, Drill, ElasticSearch, MapReduce,

Cycle Certifiant architecte BigData

CB090

Calcul et restitution, intégration	: Différentes solutions : calculs en mode batch, ou en temps réel, sur des flux de données ou des données statiques. Les produits : langage de calculs statistiques, R Statistics Language, sas, RStudio; outils de visualisation : Tableau, QlikView Ponts entre les outils statistiques et les bases BigData Outils de calcul sur des volumes importants : storm en temps réel, hadoop en mode batch. Zoom sur Hadoop : complémentarité de HDFS et MapReduce. Restitution et analyse : logstash, kibana, elk, pentaho Présentation de pig pour la conception de tâches MapReduce sur une grappe Hadoop.
Hadoop	: Les fonctionnalités du framework Hadoop. Les différentes versions. Distributions : Apache, Cloudera, Hortonworks, EMR, MapR, DSE. Spécificités de chaque distribution. Architecture et principe de fonctionnement. Terminologie : NameNode, DataNode, ResourceManager, NodeManager. Rôle des différents composants. Le projet et les modules : Hadoop Common, HDFS, YARN, Spark, MapReduce Oozie, Pig, Hive, HBase, ...
Les outils Hadoop	: Infrastructure/Mise en oeuvre : Avro, Ambari, Zookeeper, Pig, Tez, Oozie, Falcon, Pentaho Vue d'ensemble Gestion des données. Exemple de sqoop. Restitution : webhdfs, hive, Hawq, Mahout,ElasticSearch .. Outils complémentaires: Spark, SparkQL, SparkMLib, Storm, BigTop, Zebra de développement : Cascading, Scalding, Flink/Pachyderm d'analyse : RHadoop, Hama, Chukwa, kafka

Cycle Certifiant architecte BigData

CB090

Installation et configuration	: Trois modes d'installation : local, pseudo-distribué, distribué Première installation. Mise en oeuvre avec un seul noeud Hadoop. Configuration de l'environnement,étude des fichiers de configuration : core-site.xml, hdfs-site.xml, mapred-site.xml, yarn-site.xml et capacity-scheduler.xml Création des users pour les daemons hdfs et yarn,droits d'accès sur les exécutables et répertoires. Lancement des services. Démarrage des composants : hdfs, hadoop-daemon, yarn-daemon, etc .. Gestion de la grappe, différentes méthodes : ligne de commandes, API Rest, serveur http intégré, APIS natives Exemples en ligne de commandes avec hdfs, yarn, mapred Présentation des fonctions offertes par le serveur http Travaux pratiques : Organisation et configuration d'une grappe hadoop
Administration Hadoop	: Outils complémentaires à yarn et hdfs : jConsole, jconsole yarn Exemples sur le suivi de charges, l'analyse des journaux. Principe de gestion des noeuds, accès JMX. Travaux pratiques : mise en oeuvre d'un client JMX Administration HDFS : présentation des outils de stockage des fichiers, fsck, dfsadmin Mise en oeuvre sur des exemples simples de récupération de fichiers Gestion centralisée de caches avec Cacheadmin Déplacement d'un NameNode. Mise en mode maintenance.
Haute disponibilité	: Mise en place de la haute disponibilité sur une distribution Ambari. Travaux pratiques : Passage d'un système HDFS en mode HA

Cycle Certifiant architecte BigData

Sécurité	: Mécanismes de sécurité et mise en oeuvre pratique : Activation de la sécurité avec Kerberos dans core-site.xml, et dans hdfs-site.xml pour les NameNode et DataNode. Sécurisation de yarn avec la mise en oeuvre d'un proxy et d'un Linux Container Executor. Travaux pratiques : Mise en place de la sécurité Kerberos sur une distribution Ambari. Création des utilisateurs. Travaux sur les droits d'accès et les droits d'exécution. Impact au niveau des files Yarn, Oozie et Tez.
Exploitation	: Installation d'une grappe Hadoop avec Ambari. Tableau de bord. Lancement des services. Principe de la supervision des éléments par le NodeManager. Monitoring graphique avec Ambari. Présentation de Ganglia,Kibana Travaux pratiques : Visualisation des alertes en cas d'indisponibilité d'un noeud. Configuration des logs avec log4j.
NoSQL avec Cassandra	: Historique, fonctionnalités de Cassandra, licence Format des données,"key-value", traitement de volumes importants, haute disponibilité, système réparti de base de données, ...
Installation et configuration	: Prérequis. Plate-formes supportées. Etude du fichier de configuration : conf/cassandra.yaml Répertoire de travail, de stockage des données, gestion de la mémoire. Démarrage d'un noeud et test de l'interface cliente cqlsh.
CQL	: Commandes de base : connexion au système de base de données, création de colonnes,insertion, modification recherche, Le CQL : Cassandra Query Language. Exécution de scripts. Comment écrire des requêtes? Approches.

Cycle Certifiant architecte BigData

CB090

Gestion de la grappe	: Principe.Préparation du premier noeud : adresse d'écoute. Configuration de nouveaux noeuds.Notion de bootstrapping et de token. Paramètres listen_address et rpc_address. Réplication : topologie du réseau et EndpointSnitch.Stratégie de réplication. Ajout de noeuds, suppression. Cassandra dans un cloud. Mise en oeuvre avec OpenStack.
Supervision	: OpsCenter : installation, lancement. Utilisation de base. Supervision avec nodetool cfstats, ou export JMX vers des outils de supervision comme Nagios.
Exploitation	: Sauvegardes. Import/export au format JSON.
Support Hadoop	: Principe de MapReduce. Implémentation Hadoop. Mise en oeuvre depuis Cassandra.
Support Spark	: Description rapide de l'architecture spark. Mise en oeuvre depuis Cassandra. Execution de travaux Spark s'appuyant sur une grappe Cassandra.
Flux de données avec Storm	: Présentation de Storm:fonctionnalités, architecture, langages supportés Définitions:spout, bolt, topology
Architecture	: Etude des composants d'un cluster Storm : master node 'nimbus' et worker nodes Positionnement par rapport à un cluster Hadoop.Le modèle de données. Différents types de flux.
Premiers pas	: Configuration d'un environnement de développement. Installation d'un cluster Storm. Travaux pratiques sur le projet storm-starter
Flux de données	: Définition du nombre de flux dans un noeud, création de topologies regroupants des flux entre différents noeuds, communication entre flux en JSON, lecture de flux d'origines diverses (JMS, Kafka, ...)

Cycle Certifiant architecte BigData

Haute disponibilité	: Tolérance aux pannes: principe de fiabilisation des master node, workers node, nimbus Garantie de traitement des flux: principe,paramètres TOPOLOGY_MESSAGE_TIMEOUT_SECS, TOPOLOGY_ACKERS Traitements temps réel avec Trident. Scalabilité: parallèlisme dans un cluster storm, ajouts de noeuds, commande 'storm rebalance'

Filières bases de données SQL

Le Langage SQL

PostgreSQL Administration
PostgreSQL Administration avancée

MySQL Administration

Langage SQL

BD001

Durée: 3 jours
1660 €

4 au 6 février
23 au 25 avril
24 au 26 juin
1er au 3 octobre
9 au 11 décembre

Public:
Analystes, développeurs, utilisateurs, exploitants, administrateurs de bases de données.

Objectifs:
Maitriser les fonctionnalités standards du langage SQL. Connaître et maîtriser les requêtes d'interrogation SQL.

Connaissances préalables nécessaires:
Connaissance des principes des bases de données.

Programme:

Introduction au langage SQL : Le modèle relationnel, lLes composantes de SQL, les tables, la norme SQL. Le schéma général d'une base de données

Interrogations des données : La requêtes SELECT. Syntaxe générale. Sélection de lignes. L'agrégation. Le tri.
La clause WHERE, les tris avec ORDER BY, les regroupements avec GROUP BY
Les différents types de prédicats. Les expressions. Les fonctions.
Les tables temporaires. La notion de jointure : syntaxe, Inner join, Outer join
Les requêtes imbriquées : le Subselect simple, le Subselect corrélé
Les opérateurs ANY, SOME, ALL, EXISTS

Le dictionnaire des données : La définition des objets : Data Definition Language
Les types de données, la notion d'index, lLa création de tables
CREATE TABLE, CREATE INDEX, l'intégrité référentielle
Les VUES : création et utilisation

Mise à jour des données : Ajout, mise à jour ou suppression d'enregistrements avec INSERT, UPDATE, DELETE
Modification ou suppression de tables avec ALTER et DROP

Langage SQL

BD001

Les fonctions	: Présentation des fonctions les plus courantes : numériques, de test, de gestion date/heure, de manipulation des chaînes de caractères.
La confidentialité des données	: Gestion des droits d'accès, attribution et suppresssion de droits avec GRANT et REVOKE, utilisation des rôles pour sécuriser les accès
Les contraintes d'intégrité	: Intégrité contrôlée par le SGBDR : valeurs par défaut, contrôle de la valeur nulle, de l'unicité d'une colonne : DEFAULT, NOT NUL, UNIQUE, CHECK, principe de la clé primaire et contrôle par le SGBDR, notion de FOREIGN KEY
La transaction et les accès concurrents	: Principe des accès concurrents, solution des verrous, définition d'une transaction Mise en oeuvre des verrous, gestion des verrous en place sur une table Gestion des modifications : validation, retour à l'éta antérieur, synchronisation avec COMMIT, ROLLBACK, SAVEPOINT
L'optimisation	: Techniques d'optimisation des requêtes avec postgreSQL Phases d'exécution d'une requête. Analyse du plan d'exécution d'une requête EXPLAIN Bonnes pratiques et erreurs à éviter pour garantir de bonnes performances.

MySQL : Administration

BD012

Durée: 3 jours
1660 €

11 au 13 février
13 au 15 mai
16 au 18 septembre
12 au 14 novembre

Public:

Toute personne souhaitant configurer, installer et exploiter une base de données MySQL

Objectifs:

Comprendre le fonctionnement, et savoir installer, configurer et administrer une base de données MySQL (le cours est réalisé sur une version 5 de MySQL).

Connaissances préalables nécessaires:

Il est demandé aux participants de connaître les notions de base sur SQL.

Programme:

Introduction : Présentation, historique, les versions MySQL (standard, Max, Pro, Classic), les licences (GPL et commerciale).
Les composants du serveur MySQL.Caractéristiques: transactions, clusters.

Installation : Choix du produit à installer : les RPMs, le code compilé ou les sources. Installation, configuration.
Les scripts fournis avec MySQL : démarrage du serveur, création des tables de droits d'accès, démarrage de multi-serveurs, ...
Outils graphiques.

Sécurité : Système des privilèges :
principe de fonctionnement, authentification, contrôle des droits pour les requêtes
Gestion des comptes utilisateurs : création/suppression de comptes, limitation des ressources, sécurisation des accès
Mise en place de SSL.

Les fichiers de logs : Les erreurs, les modifications du fichier ISAM, les requêtes

MySQL : Administration

Sauvegardes : Les tables MyISAM et InnoDB
Utilitaire myisamchk : contrôler, réparer, optimiser. Vérification sur base à l'arrêt. Réparation.
Vérification/réparation en cours d'exploitation.Méthode de sauvegarde des données MySQL, script mysqldump, ou mysqlhotcopy

PostgreSQL : Administration

BD021

Durée: 3 jours
1660 €

25 au 27 février — 9 au 11 septembre
20 au 22 mai — 25 au 27 novembre

Public:

Toute personne souhaitant configurer, installer et exploiter une base de données PostgreSQL

Objectifs:

Comprendre le fonctionnement, et savoir installer, configurer et exploiter une base de données PostgreSQL.

Connaissances préalables nécessaires:

Il est demandé aux participants de connaître les notions de base sur SQL.

Programme:

Introduction : Présentation, historique, les versions PostgreSQL, les outils complémentaires et les licences.
Les composants du serveur PostgreSQL: serveur, client, connecteurs jdbc, tcl, pl, python

Installation : Choix du produit à installer : les RPMs, le code compilé ou les sources.
Installation. Configuration, organisation du répertoire /var/lib/pgsql, fichier postgresql.conf.
Les scripts fournis avec PostgreSQL : démarrage du serveur, création des tables de droits d'accès, démarrage de multi-serveurs, ...

Sécurité : Système des privilèges : principe de fonctionnement, authentification, contrôle des droits pour les requêtes
Gestion des comptes utilisateurs : création/suppression de comptes, limitation des ressources, sécurisation des accès : fichier pg_hba.conf
Mapping avec les utilisateurs systèmes: pg_ident.conf

Utilisation : Commande psql. Accès aux tables. Les commandes en \. pgadmin : installation, configuration.

BD021

PostgreSQL : Administration

Exploitation : Sauvegardes/Restaurations : contrôler l'état de la table, réparer, optimiser :
pg_dump, pg_dumpall, vacuumdb.Méthode de sauvegarde des données PostgreSQL
Montée de niveaux : copie de serveur à serveur, réplication.

PostgreSQL :administration avancée

BD022

Durée: 2 jours
1090 €

28 février au 1er mars
23 au 24 mai
12 au 13 septembre
28 au 29 novembre

Public:

Administrateurs souhaitant approfondir leurs connaissances de l'administration de postgreSQL

Objectifs:

Savoir configurer les sauvegardes et l'archivage, répondre aux contraintes de haute disponibilité, mettre en oeuvre la réplication,

Connaissances préalables nécessaires:

Il est demandé aux participants de connaître l'administration postgreSQL de base.

Programme:

Sauvegardes et archivage : Les différentes méthodes et outils : sauvegarde SQL, système de fichiers, archivage continus.
pgdump : principe, exemple de sauvegarde et restauration des données avec psql,
pgdumpall : sauvegarde de toutes les bases d'une instance
Archivage continu avec WAL. Principe, configuration de l'archivage WAL
Sauvegardes avec pg_basebackup. Configuration de la récupération d'un archivage continu

Haute disponibilité : Différentes méthodes. Principe des serveurs warm et hot standby.
Utilisation des flux WAL. Mise en oeuvre du transfert de journaux et de la réplication en continu (streaming replication)

Optimisation : Outils de supervision de l'activité de la base de données
Configuration des statistiques:
paramètres : track_activities, track_count, track_functions, track_io_timing. Contrôle des verrous : pg_locks

Filières Systèmes Unix/Linux

Linux/Unix Introduction
Le Shell
Shell avancé
Administration Linux
Administration avancée Linux
Services réseaux sous Linux

Haute disponibilité sous Linux
Linux système sécurisé
Linux sécurité des accès
Linux optimisation et performances

Virtualisation Linux
Virtualisation avec Xen
Virtualisation avec kvm
Virtualisation avec lxc

Linux/unix introduction

UX100

Durée: 3 jours
1640 €

21 au 23 janvier
25 au 27 mars
27 au 29 mai
23 au 25 septembre
18 au 20 novembre

Public:
Utilisateurs, exploitants de systèmes Unix/Linux.

Objectifs:
Connaître les principes de fonctionnement du système Unix/Linux.

Connaissances préalables nécessaires:
Connaissances générales en informatique.

Programme:

Introduction : Présentation de Linux et Unix, définitions de base, version de noyau, distributions.
Méthodes pour obtenir de l'information.
Rappel rapide sur l'organisation d'un système.
Arrêt/relance du système (shutdown, halt, reboot, sync).
Commandes de base.
Aide en ligne.

Interface graphique : Présentation : startx.
Lancement d'applications, modification des menus, mini-applications d'interfaces
Présentation de Gnome, KDE, WindowMaker, XFCE
Travaux pratiques :
configuration de l'interface graphique,
lancement d'applications, modification des menus

Connexion : Principe de la connexion/déconnexion,
les commandes en arrière-plan,
les redirections (entree standard/sortie standard)
Travaux pratiques :
lancement de commande en arrière-plan,
mise en oeuvre des "pipes".

UX100

Linux/unix introduction

Les fichiers	: Organisation des données sur un serveur Unix Structure des disques Le système de fichiers Les types de fichiers. Chemin d'accès et nom de fichier. Manipulations de fichiers et de répertoires Recherche de fichiers : la commande find
Les filtres	: Le mécanisme des tubes Exemples de commandes filtre Grep et expressions régulières
La sécurité	: Le fichier des utilisateurs et le fichier des groupes Le mode d'un fichier. Modification des permissions.
Les processus	: Gestion de la mémoire et des processus. Caractéristiques d'un processus Processus en arrière-plan. Les travaux batch.
Commandes shell standard	: Commandes classiques : l'aide en ligne avec man, l'arborescence : pwd, cd, mkdir; rmdir, ls, du, file, manipulations de fichiers : cp, rm, mv, find, grep, ln, cat, more, ... gestion des processus : ps, kill, date, who, commandes d'environnement : tty, id, passwd, lpr, env, . Travaux pratiques : mise en oeuvre des commandes étudiées. Ajout de logiciels. Utilisation de périphériques : lsmod, insmod, lspci Impression : lpr, configuration. Montage de cdrom : mount, umount Configuration de /etc/fstab
Applications	: Présentation des applications courantes: bureautique, dessins, utilisateur WEB, réseau, et des applications serveurs.

Le Shell

UX002

Durée: 2 jours
1110 €

24 au 25 janvier
28 au 29 mars
3 au 4 juin
26 au 27 septembre
21 au 22 novembre

Public:
Les développeurs d'applications sur UNIX, analystes d'exploitation, exploitants et administrateurs.

Objectifs:
Maîtriser la programmation en bourne Shell.

Connaissances préalables nécessaires:
Connaissance des principes de base du système UNIX.

Programme:

Généralités	: Présentation du shell : interpréteur de commande Unix. Modes d'exécution d'un script. Les alias. Les méta-caractères.
Les variables	: Portée des variables : locales, globales, environnement Les différents types de variables. Définir et manipuler des variables. Transmission de paramètres.
Fichiers d'environnement	: Fichier d'initialisation général : .profile Fichier d'initialisation local : .kshrc
Les entrées/sorties	: Accès en lecture/ écriture.
Structures de contrôle	: Les instructions test et expr. Expressions conditionnelles Gestion des boucles Boucles for, while, until Tests 'if', tri avec 'case' Sortie de boucles avec break, continue et exit

UX004

Atelier : Shell avance

Durée: 3 jours
1680 €

25 au 27février — 24 au 26 juin
23 au 25 avril — 16 au 18 septembre

Public:
Les développeurs d'applications sur UNIX, et personnes chargées de la mise en production des applications, les exploitants, les administrateurs.

Objectifs:
Appliquer les techniques avancées du shell. Connaître les outils disponibles et les bonnes pratiques concernant la programmation de scripts.Ce stage est réalisé sous la forme d'un atelier de travaux pratiques

Connaissances préalables nécessaires:
Connaissance des principes de base du système UNIX/Linux et de la programmation shell de base.

Programme:

Rappels techniques : Différents modes d'exécution des scripts : nohup, exec, at, ... Interruption des scripts : les signaux. Le suivi de consommation : time, eval : réinterprétation d'une commande, select : gestion de menus (ksh),getopts : décoder les options d'un script
Programmation parallèle, programmation événementielle.Gestion des alertes. Le debugging et l'optimisation.

Programmation m4 : utilisation de la protection, encapsulation

Les outils : grep, awk, sed.Les variables disponibles avec awk,; les fonctions.
opérateurs, le contrôle d'exécution.Utilisation des expressions régulières dans sed.Les sous-expressions

UX004

Atelier : Shell avance

Mise en oeuvre : Développement en shell d'un superviseur Unix/Linux et réseau
Fonctionnalités : console centrale de supervision.Visualisation de l'état de chaque poste supervisé.Journalisation des opérations et états.
Gestion des utilisateurs, ressources (mémoire, cpu, disques).Supervision de processus.

Administration Linux

UX111

Durée: 5 jours
2440 €

4 au 8 février
15 au 19 avril
17 au 23 juin
9 au 13 septembre
25 au 29 novembre

Public:

Administrateurs, et toute personne souhaitant maîtriser l'installation, la configuration d'un système Linux.

Objectifs:

Savoir installer, administrer un système Linux. Chaque participant dispose des différentes distributions (Debian, Redhat) et peut, s'il le souhaite, tester les travaux pratiques sur le système de son choix.

Connaissances préalables nécessaires:

Des connaissances de base des systèmes Unix et/ou Linux sont nécessaires, ainsi que du Shell.

Programme:

Introduction : Linux et l'opensource : historique, caractéristiques de linux
Les distributions, les différences et points communs.
Rappel rapide sur l'organisation d'un système.
Arrêt/relance du système (shutdown, halt, reboot, sync).
Les apports de systemd

Installation : Les phases d'installation d'un système Linux.Options dans les chargeurs : grub. Les outils d'installation. Gestions de paquets. Les différentes méthodes.
RPM, le système RedHat : historique, présentation et fonctionnement de la commande rpm, principales options pour l'installation, l'interrogation, l'affichage du contenu d'un paquet...
Travaux pratiques : requêtes d'interrogation des packages rpm, installation et mise à jour de packages.
Le paquetage Debian : fonctionnalités, format et statut des paquetages,les applications de gestion (dpkg, dpkg-deb, dpkg-query, apt, ...)
Travaux pratiques avec dpkg : extraction des informations concernant un paquet.Présentation des outils : apt, yum, urpmi.

UX111

Administration Linux

Environnement graphique	: Présentation, gestionnaire de fenêtres. Différentes solutions : gnome, KDE, Windowmaker, xfce.
Outils d'administration	: Webmin : Présentation, installation, configuration.Démonstration.
Systèmes de fichiers	: Définitions : inodes, filesystem, partition Organisation, gestion et maintenance : utilisation de la commande mkfs. Principe du montage d'un périphérique. Travaux pratiques : mise en place d'un montage à l'initialisation du système (/etc/fstab) et d'un montage temporaire (commande mount). Exploitation et maintien de l'intégrité des systèmes de fichiers : commandes mkfs, mount, umount, df. Travaux pratiques : comparer le résultat des commandes df et du Test de montage d'un système de fichiers sur un point d'ancrage non vide. Présentation de différents types de systèmes de fichiers : ext3, reiserFs, xfs, jfs.Les autres systèmes de fichiers : fat, vfat, nfs, smb. Partition : création d'images de partitions. Synchronisation de données. Chiffrement des données.
Utilisateurs	: Etude des fichiers /etc/passwd, /etc/group, /etc/shadow. Gestion des comptes utilisateurs: useradd, usermod, userdel, passwd, gestion des groupes : groupadd, groupdel, ajout d'utilisateurs,création d'administrateurs de groupes, droits d'accès, politique d'accès. Travaux pratiques : création d'utilisateurs et de groupes, puis vérification de cohérence avec la commande pwck. Contrôle des connexions de root : les objectifs et les méthodes. Travaux pratiques : utilisation de l'outil "john the ripper" pour la recherche de mots de passe. Introduction à PAM : Pluggable Authentication Modules.
Processus	: Les processus. Les threads. Gestion des priorités.Utilisation des pseudo-processus /proc: stat, cpuinfo...

Administration Linux

UX111

Sécurité des données	: Sauvegardes Outils sauvegarde/archivage/compression : gzip, zip, tar, dd, cpio, dump, restore. Sauvegarde du système, création de CD de secours.Travaux pratiques : sauvegarde par cpio, réalisation d'un archivage par tar. Tests de restauration des données. Synchronisation des données par rsync sur des serveurs distants.
Impressions	: Les services d'impression, démarrage/arrêt des services d'impression. Présentation de CUPS : Common Unix Printing SystemDéfinitions : classes d'imprimantes, classes implicites, destination, filtres, backends. Installation d'une imprimante, modification d'un pilote : lpr, cups, printtool, system-config-printer.
Programmation de tâches	: Le besoin, l'automatisation des tâches système.Exécution différée avec at. Programmation de tâches avec cron. Etude du fichier crontab.Les produits du marché : openPBS, fcron
Exploitation	: Journaux : /var/log/messages
Réseau IP	: Les objets à configurer : les interfaces réseaux, les routes, le DNS.Principe de la configuration dynamique ou statique. Configuration, nommage/activation des interfaces réseau, drivers.Etude des fichiers /etc/hosts, /etc/nsswitch, /etc/resolv.conf. Travaux pratiques : création d'une interface réseau, visualisation, configuration de plusieurs adresses IP sur la même interface physique. Ajout d'une route, d'un hôte, d'un serveur DNS, et tests.Présentation des utilitaires ssh, clients windows (Putty, WinSCP)

Administration Linux

UX111

NFS	: Fonctionnalités : partage de fichiers en réseau, avec gestion de la sécurité. Description du fonctionnement client/serveur.Etude du fichier /etc/exports. Travaux pratiques : configuration d'un serveur NFS sur chaque poste, et configuration des clients NFS pour tester les accès.
Intégration système d'information	: Samba : Principe. Intégration de SMB au niveau des couches réseaux.Fonctionnalités : partage de répertoires, d'imprimantes, création de comptes.... Travaux pratiques : Installation et configuration de samba pour le partage de fichiers.

Administration avancée Linux

UX140

Durée: 5 jours
2480 €

18 au 22 février
15 au 19 avril
17 au 21 juin
23 au 27 septembre
2 au 6 décembre

Public:

Administrateurs, et toute personne souhaitant approfondir l'administration d'un système Linux.

Objectifs:

Savoir installer, administrer, faire évoluer une distribution. Ce cours a lieu sur Linux RedHat, et sur Debian pour la partie "apt". Il est essentiellement basé sur des travaux pratiques.

Connaissances préalables nécessaires:

Connaître les techniques d'administration d'un système Unix ou Linux.

Programme:

Distribution : Présentation : RedHat Package Manager.
Les distributions qui utilisent les rpm.
Fonctionnalités : sécurité, méta-données, gestion des dépendance.
Détails de la commande rpm, et de ses options.
Travaux pratiques :
mise en oeuvre, installation, désisntallation, requêtes documentation.
Construction de RPMs : depuis les sources jusqu'au package.
Description des paquets DEB : fonctionnement apt, dpkg, dselect, debconf.
L'outil apt : principe, les répertoires apt, fichiers release.
Les commandes apt-get, apt-cache.
Les frontaux apt : apt-shell, aptitude, synaptic.
Travaux pratiques :
recherche d'informations sur un paquet,
installation d'une mise à jour.

Administration avancée Linux

UX140

Démarrage/Installation	: Analyse du mode de démarrage : grub, Anaconda Le système kickstart.Analyse d'une image initrd. Travaux pratiques : Modification d'un initrd, ajout de modules. Création de média d'installation. Boot sur un périphérique USB depuis un CD.
Systèmes de fichiers journalisés	: Exemples de systèmes de fichiers journalisés. Les types de journalisation. XFS : fonctionnement, mise en oeuvre, administration compatibilité NFS Ext3, ext4 : caractéristiques et mise en oeuvre.
LVM	: Logical Volume Manager. Présentation. Définitions : VFS, EVMS, Volumes physiques, groupes de volumes, volumes logiques, extension logique. Travaux pratiques :mise en place de partitions LVM. Formatage en xfs. Mode d'utilisation des LVM : les snapshots, le redimensionnement, la concaténation de groupes de volumes. Exercice : création de volumes physiques, de groupes de volumes, création de snapshot. Ajout d'un disque, sauvegarde d'une partition, redimensionnement.
RAID	: Définitions. Les principaux types de RAID. Le RAID Logiciel sous Linux : présentation, outils d'administration. Travaux pratiques : utilisation des outils madm pour créer un système de fichiers RAID. Mise en évidence des reprises sur incidents : simulation de panne, synchronisation des données. Analyse des performances.

Administration avancée Linux

UX140

Authentification en production	: Besoin de mécanismes d'authentification performants et fiables. pam : gestion des modules d'authentification. Principe de base. Travaux pratiques : configuration, mise en oeuvre. Les modules : access, chroot, cracklib, etc ... Ldap : Lightweight Directory Access Protocol Les modèles, la conception d'une arborescence. Interface pam/ldap. Travaux pratiques : mise en oeuvre avec Openldap et l'automonteur
Performances	: Le besoin, les points à surveiller. Les points de mesures : utilisation CPU, occupation des disques, charge réseau, occupation mémoire, etc ... Commandes de suivi des ressources processeurs et mémoire : vmstat, top. Commandes de suivi des ressources réseaux : netstat, ntop, iptraf. Surveillance des ressources disques : df, lsof Gestion de la fragmentation, pagination. Travaux pratiques :analyse des informations de /proc/stat, /proc/cpuinfo et de l'accounting. Les outils : oprofile, dtstat, systat.
Ressources	: Les quotas disques : principe, mise en place. Travaux pratiques : déclaration des quotas dans le fichier /etc/fstabb, activation des quotas, exemple de dépassement de limite d'espace disque autorisé.
Noyau	: Compilation du noyau : présentation, les différentes phases. Travaux pratiques :téléchargement et décompression des sources configuratin avec make, recompilation.
Périphériques	: Périphériques non standards. Installation de modules: modprobe, insmod. Le répertoire hotplug.

Les services réseaux Linux

UX112

Durée: 4 jours

Prix et dates: nous consulter

Public:
Administrateurs systèmes et réseaux.

Objectifs:
Savoir installer, configurer et sécuriser les principaux services réseaux sur Linux.

Connaissances préalables nécessaires:
Des notions de base sur le système Unix ainsi que sur TCP/IP sont souhaitées.

Programme:

Configuration IP : Présentation.
Activation du réseau.
Interfaces réseau.
Routage.
Fichiers de configuration.

Outils réseau : Outils de trace
Tcpdump
Outils de diagnostic

Les services : Serveurs de configuration
dns, dhcp, bootp (présentation de kickstart
Serveurs de fichiers : nfs, ftp, tftp, http
Serveurs d'accès : routage, firewall, proxy

DHCP : Définition, principe.
Configuration poste client, serveur.
Notion de bail.
configuration avancée.
Redondance DHCP.

DNS : Définition, fonctionnement.
Travaux pratiques avec bind : configuration client et serveur.

UX112

Les services réseaux Linux

FTP	: File transfer protocol Travaux pratiques : mise en oeuvre de ftp: configuration, droits sur les répertoires, gestion des utilisateurs, surveillance, fichiers de logs
NTP	: Définition.Fonctionnement. Déclaration d'un point de synchronisation. Configuration d'un serveur d'horloge. Configuration de clients. Architecture. Contrôles d'accès. Implémentations de NTP.
Messagerie	: Les protocoles, POP3 et IMAP4 Structure des messages. Présentation de Postfix, installation et configuration. Fichiers master.cf, main.cf
Serveur Web	: Installation Apache : configuration de base, configuration multi-sites (httpd.conf) Suivi : access_log, error_log Principe des scripts CGI, et des modules.les hôtes virtuels
NFS	: présentation, fonctionnement, configuration d'un serveur NFS et des postes clients partage de fichiers Les groupes de confiance, méthodes de protection.
Intégration hétérogène	: interconnexion Unix/Windows : samba : configuration et installation accès depuis des clients windows, et Linux en mode texte : smbclient, smb4K : le navigateur samba
Sécurisation des accès réseau	: Connexion directe, distante, liste des points d'entrée dans le système. Analyse des fichiers journaux du réseau Vérification de l'intégrité du système à l'aide des outils : tcpdump, sniffit, cop, satan ping, traceroute
Administration distante	: Webmin : installation, présentation de l'interface et des fonctionnalités de webmin

Haute disponibilité Linux

UX115

Durée: 3 jours
1710 €

11 au 13 février
27 au 29 mai
1er au 3 juillet
12 au 14 novembre

Public:

Administrateurs Linux, ou toute personne souhaitant mettre en oeuvre un système Linux avec des contraintes de haute disponibilité.

Objectifs:

Connaître et savoir mettre en oeuvre les mécanismes disponibles sur Linux pour offrir un service continu.

Connaissances préalables nécessaires:

Une bonne connaissance d'un système Unix et des réseaux IP est nécessaire. Des notions d'administration sont souhaitées.

Programme:

Introduction : Le besoin : pourquoi la haute disponibilité, mesure de la disponibilité.
Quelques définitions : tolérance aux pannes, fail-over, RAID, Mirroring, redondance, MTBF, etc ...
Les acteurs du marché, positionnement de Linux.
Présentation de l'architecture LVS.
Les solutions de haute disponibilité.

Clustering : Les différentes fonctions de clustering :
répartition des accès disques, répartition de la charge CPU,
basculement automatique ou programmé sur un autre processeur,
exécution simultanée sur plusieurs processeurs.

Adresses réseaux : Principe du basculement d'adresses.
Solution avec Fake.
Agrégation d'interfaces réseau.
Travaux pratiques :
mise en place de l'agrégation avex deux cartes réseaux ethernet.
Configuration dynamique et configuration statique.
Test et vérification dans les fichiers journaux.

Haute disponibilité Linux

UX115

Linux Virtual Server : Architecture : pacemaker, ldirector,heartbeat, fake, coda
Utilisation de mon pour la détection des services défaillants.
Travaux pratiques :
Installation, configuration de heartbeat et ldirectord
Configuration de Pacemaker pour la gestion du cluster.

IPVS : Présentation : IP Virtual Server.
Répartition de charge.
Contrainte au niveau du noyau.
Travaux pratiques :
préparation d'un noyau IPVS, configuration passerelle.
Mise en place d'un cluster.

ldirectord : Présentation : Linux director daemon.
Fonctionnalités.
Travaux pratiques :
installation et configuration de ldirectord

Applications : Intégration LVS avec Keepalived.
Architecture, prérequis du noyau.
Travaux pratiques :
Installation et configuration keepalived.
Gestion de ressources avec Pacemaker.
Présentation de la RedHat Cluster Suite.
Répartition de requêtes http, gestion des sticky session.
Répartition de charges, routage de niveau 7.
Présentation des solutions WebSphere, JBoss et Jonas.
Travaux pratiques :
Mise en oeuvre du répartiteur de charge HAProxy en mode HTTP.

Données : Le besoin, les différentes solutions techniques :
réplication de données en réseau, ou en local.
Exemples de Coda, Logical Volume Manager.
Le RAID, RAID logiciel sous Linux : raidtool, mdadm.
Les systèmes de fichiers haute disponibilité :
DRDB (Distributed Replicated Block Device)
Fonctionnalités, installation et configuration.
Cluster Active/Hot standby avec ext3.
Cluster Active/Active avec gfs.
Export de gfs par gndb.
Intégration avec heartbeat.

Linux système sécurisé

UX117

Durée: 3 jours
1710 €

18 au 20 février — 16 au 18 septembre
13 au 15 mai — 18 au 20 novembre

Public:
Toute personne souhaitant mettre en place une sécurité optimale sur un système Linux, et plus particulièrement les administrateurs système et sécurité.

Objectifs:
Savoir configurer les mécanismes de sécurité de Linux.

Connaissances préalables nécessaires:
Une bonne connaissance de l'administration des systèmes Unix/Linux est nécessaire.

Programme:

Introduction : Le besoin, définition du D.I.C.
Les attaques possibles.
Evaluation des risques.
Méthodes de protection.

Gestion utilisateurs : Rappels sur les notions de base de sécurité sur Unix :
modes d'accès, comptes utilisateurs, groupes, utilisateurs génériques de gestion de ressources.
Fichiers /etc/passwd, /etc/group, /etc/shadow.Codage des mots de passe.Création, modification, suppression de comptes utilisateurs.
Gestion des groupes :ajout, retrait d'utilisateurs, création d'administrateurs de groupes.
Affectation d'un mot de passe au groupe.Vérification de cohérence : pwck.Connexions du compte root, contrôle de connexions.
Outil de recherche de mots de passe.
Travaux pratiques : installation et mise en oeuvre de l'outil "John the ripper" en mode "single-crack".
Prise de privilèges : sudo, sudoers.

Linux système sécurisé

UX117

Authentification : pam: gestion des modules d'authentification. Présentation et exemples d'utilisation.Principe de base, configuration.Les modules : différents types de modules (auth, account, session, password).
Notion de pile de modules.
Travaux pratiques :mise en oeuvre de PAM et de quelques modules parmi les plus courants :access, chroot, cracklib, env, ftp, groups, limits, listfile, mkhomedir, tally, time, unix, wheel

Sécurisation traitements : Les risques : le déni de service, exemples de virus sur un système Linux.
Travaux pratiques : exploitation d'un débordement de pile.Les moyens de détection, la surveillance, les traces : syslog, l'accounting. L'audit de securité.Méthodes de protection : démarche sur les systèmes Linux.

Sécurité du noyau : Les différentes approches de sécurisation du noyau.
Présentation de GrSecurity et SELinux. Travaux pratiques avec GrSecurity :installation, configuration du noyau, paramétrage du niveau de sécurité.
Administration avec grAdm2.Génération d'une politique : learning mode.Mise en place des règles d'ACL.
L'ACL GrSec.Restrictions d'accès aux appels systèmes. Masquage de processus.Visibilité du répertoire /proc.
Restrictions chroot. SELinux : principe, configuration du noyau, options du noyau.
Travaux pratiques :définition d'une politique de sécurité.Installation et activation de la politique de sécurité dans le fichier /etc/selinux/config.

Sécurité des données : Contrôle de la cohérence du système de fichiers : fsck.Procédure de vérification.Sauvegardes : définitions
Commandes et outils standards.Utilisation des sauvegardes pour la disponibilité des données. Outils sauvegarde/archivage/compression :
gzip, zip, tar, dump, restore, dd, cpio, rsyncChiffrement des disques durs :mise en oeuvre de LUKS et dm-crypt
Protection de la mémoire :principe et outils de sécurisation.

UX117

Linux système sécurisé

Sécurité système de fichiers	Sécurité: mise en place des contrôles d'accès .ACL : principe des listes de contrôle d'accès POSIX. Travaux pratiques : mise en place des ACL sur xfs. Les quotas : principe, mise en place dans le fichier /etc/fstab. La commande edquota pour l'édition, et le paramétrage, et la commande quota pour la visualisation. Travaux pratiques : mise en place des quotas

Linux sécurité des accès

UX118

Durée: 3 jours
1710 €

30 janvier au 1er février
27 au 29 mai
23 au 25 septembre
18 au 20 novembre

Public:
Toute personne souhaitant sécuriser les accès à un système Linux

Objectifs:
Savoir configurer les mécanismes de sécurité réseau de Linux.

Connaissances préalables nécessaires:
Une bonne connaissance de l'administration des systèmes Unix/Linux et des réseaux TCP/IP est nécessaire.

Programme:

Introduction	: Le besoin, définition du D.I.C.Les attaques possibles.Evaluation des risques.Méthodes de protection.
Les ports de niveaux 5	: Rappels sur la notion de port.Les ports UDP et les ports liés au réseau.Exemples de trames.
Outils de captures réseau	: Les analyseurs de trames : tcpdump, wireshark.Travaux pratiques :mise en oeuvre de tcpdump, options usuelles, et possibilités de filtrage. Installation de Wireshark, capture et analyse de paquets.
Outils de Diagnostic	: Scanners de ports, outils d'audit externe et d'audit interne.Exemples de nmap, hping, sniffit...
Audit réseau	: OpenVAS (Opensource Vulnerability Assessment Scanner) : principe de fonctionnement, installation. Travaux pratiques :réalisation d'un audit réseau avec openVAS.

UX118

Linux sécurité des accès

Sécurisation des accès réseau	:	Protection de services réseaux au travers de xinetd. Les tcp-wrappers: telnet, tftp, snmp, ftp, pop3s, imap4s Les contrôles d'accès :Etude des fichiers /etc/hosts.allow et /etc/hosts.deny Les accès réseaux : sftp, les r-commandes (rlogin, rsh).Sécurisation des transferts de fichiers avec vsftp Présentation d'openSSH.Travaux pratiques :configuration du serveur et du client pour la mise en place d'un tunnel X11 et ssh. Sécurisation http (apache) :lors de l'exécution des processus (directives user et group), portée des balises,restriction d'accès par méthode : balise Limit, LimitExcept,le fichier .htaccess : autorisation ou restriction d'accès. Authentification HTTP.Création d'utilisateurs avec htpasswd.
VPN , tunnels, iptables	:	Définitions : DMZ, coupe-feux, proxy. VPN et tunnels.Principe de fonctionnement. Présentation des tunnels chiffrés.Travaux pratiques :mise en oeuvre de stunnel pour sécuriser une messagerie smtp. Présentation d'openVPN.Travaux pratiques :installation, configuration, tests de connexion,création d'u ntunnel sécurisé par clé statique.Certificats : SERV et CLT. Pare-feux : les iptables, le filtrage de paquets,définition d'une politique de sécurité. Travaux pratiques :mise en place des iptables.Traduction d'adresse, traduction de ports.Architecture avec pare-feux et tunneling.
Proxy Squid	:	Présentation, principe de fonctionnement.Architecture, hiérarchie de serveurs cache.Exemple d'utilisation, systèmes d'exploitation concernés, logiciels complémentaires. Mécanismes de configuration manuelle, automatique. Scripts d'auto-configuration, filtrage suivant DNS, par protocole. Clients en mode texte,robots. Installation dans le navigateur.Principe et syntaxe des ACL. Optimisation de l'utilisation du serveur. Restriction d'accès par hôte, par réseau, par plage horaire, par jour, par site. Mise en cache des données. Méthodes d'authentification.

Linux : optimisation performances métrologie

UX119

Durée: 2 jours
1205 €

7 au 8 mars
6 au 7 juin
5 au 6 septembre
14 au 15 novembre

Public:
Administrateurs, et toute personne souhaitant connaître les éléments permettant d'améliorer les performances d'un système Linux.

Objectifs:
Connaître les points du système à mesurer. Comprendre leur impact sur les performances globales du système et savoir les adapter à un mode de fonctionnement (client, serveur, station, base de données, messagerie...)

Connaissances préalables nécessaires:
Une bonne connaissance d'un système Linux est nécessaire. Des notions d'administration sont souhaitées.

Programme:

Introduction : Qu'est ce que la gestion des performances?

Mesures : Les éléments à prendre en compte, les points de mesures.
Utilisation des pseudo-systèmes /proc et /sys: stat, cpuinfo, ...
Utilisation des processus système: kswapd, swpctl, rsyslogd
Commandes : vmstat, lscpu, chcpu
TP : création d'un utilitaire d'extraction des informations système.

Outils : Présentation des outils oprofile, sysstat, dstat, tuned, tuned-adm
TP avec oprofile.

Systèmes de fichiers : Les différents types de systèmes de fichiers. Les systèmes natifs : ext2, ext3, ext4, xfs,
Gestion de la fragmentation, pagination.
Les systèmes émulés : vfat, ntfs. Les systèmes distribués : nfs, cifs
Options : rsize, wsize, timeo, retrans,...
TP : outil de mesure des accès.

UX119

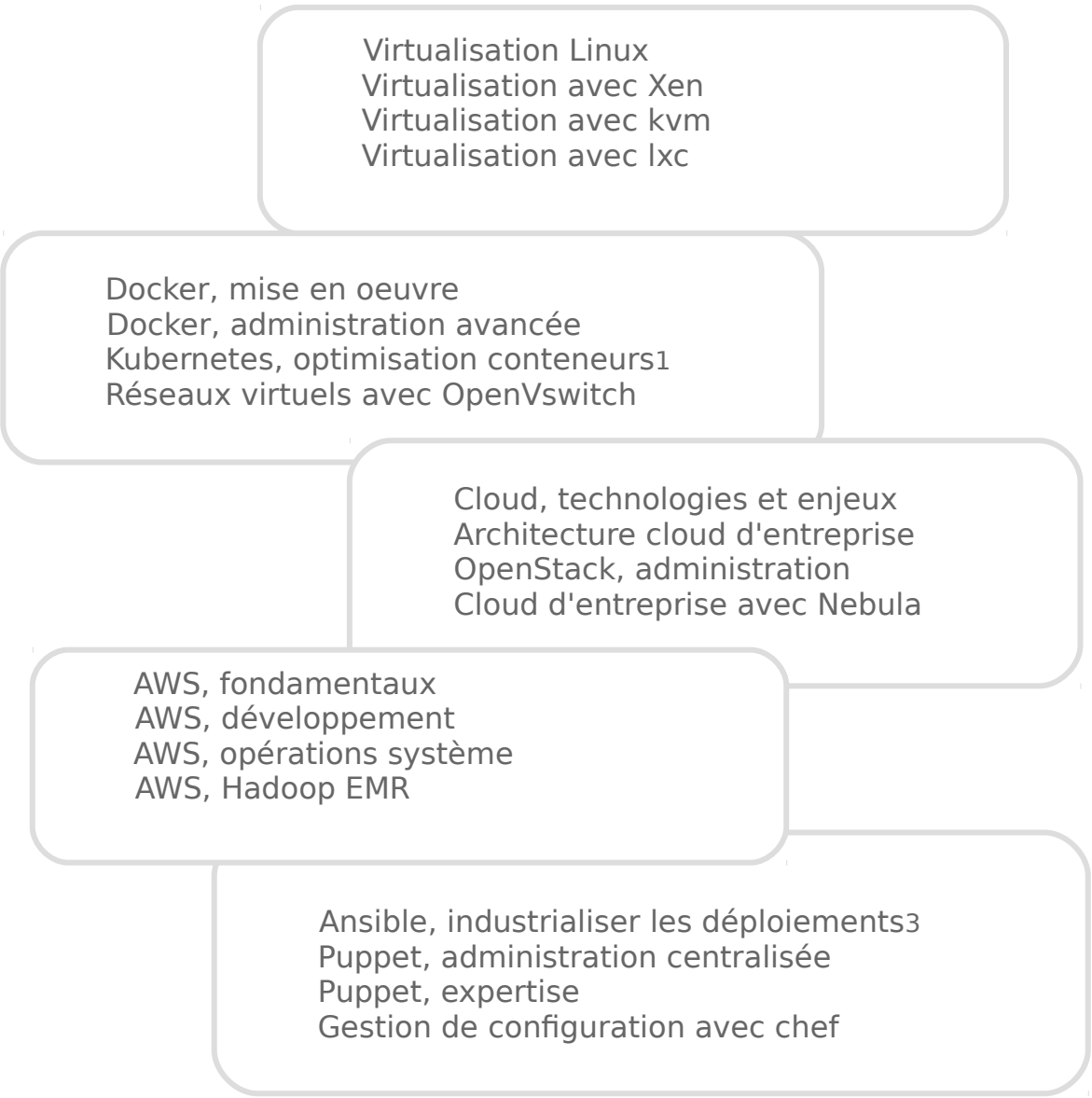
Linux : optimisation performances métrologie

Utilisateurs : accounting, quotas, fichiers de logs. ulimit, prlimit.

Réseau : Commandes : iptraf, nstat, rtacct
Exploitation des éléments statistiques produits
Gestion des ressources, qos avec tc, ifstat.
Outils
mrtg, rrdtool, SystemTap, DTrace, Phoronix test suite, TSung

Cgroups : Gestion des performances. Limitations des ressources affectées à un ou plusieurs processus.
Introduction au cloisenement.

Filières virtualisation, cloud et orchestration



Virtualisation Linux

SY011

Durée: 2 jours

Prix et dates: nous consulter

Public:
Chefs de projet, administrateurs souhaitant mettre en oeuvre une solution de virtualisation Linux.

Objectifs:
Connaître les différentes solutions de virtualisation sur Linux, et leurs caractéristiques.

Connaissances préalables nécessaires:
Une bonne connaissance du système Unix/Linux est nécessaire.

Programme:

Introduction	: Objectifs d'un système d'exploitation, gestion de ressources. Partager des ressources entre plusieurs applications, systèmes... Notion de virtualisation, quelle granularité (disques, système d'exploitation, machines physiques...) Historique : VM (Virtual Machine), VMWare, UML, Xen...
Les différentes techniques de virtualisation possibles sur Linux	: conteneurs d'application, noyaux secondaires, machines virtuelles, hyperviseur, virtualisation matérielle...
Xen	: Présentation de l'architecture de virtualisation Xen.Compilation d'un noyau Xen. Gestion des domaines :Création d'un domaine, arrêt d'un domaine. Console d'administration.
VirtualBox	: Principe et caractéristiques du produit. Les différentes éditions.Travaux pratiques : installation VirtualBox.Création de machines virtuelles.Différents paramètres de configuration.Configuration des machines virtuelles en XML.

Virtualisation Linux

lxc	: Présentation des Linux Containers. Objectifs du projet . Isolation et contrôle des ressources. Principe des 'cgroup' et création de containers. Travaux pratiques :activation des cgroup, installation lxc. Utilisation de lxc-checkconfig.Configuration de containers. Exemple de Busybox
QEMU et kvm	: Principe de QEMU et architecture. Travaux pratiques : installation et lancement d'une image Etude des options de lancement de qemu.Consoles des machines virtuelles : graphiques (console VNC, Spice, ..) ou consoles en mode texte. Kernel Based Virtual Machine : positionnement par rapport aux autres systèmes de virtualisation, et par rapport à QEMU Travaux pratiques : installation avec un noyau contenant les modules kvm. Gestion des images :création d'images, différents supports possibles, options de lancement Travaux pratiques :commandes info, check, resize, convert.Gestion du matériel:architectures supportées, processeurs, mémoire, périphériques de stockage, audio, video, usb, bluetooth, etc ... Configuration du réseau:différents modes possibles (user, tap, bridge,...) Travaux pratiques :configuration réseau sur les images créées aux chapitres précédents Snapshots et migrations :principe de fonctionnement .Mise en oeuvre et options de la commande 'migrate'
Administration avec libvirt	: Présentation de l'API libvirt et des fonctionnalités apportées,Virtual Machine Manager Travaux pratiques :installation de libvirt et lancement de virt-manager

Virtualisation avec Xen

SY004

Durée: 2 jours

Prix et dates: nous consulter

Public:

Toute personne souhaitant mettre en place la virtualisation avec Xen, administrateurs, exploitants.

Objectifs:

Comprendre les principes de la solution de virtualisation Xen. Savoir configurer et installer Xen.

Connaissances préalables nécessaires:

Une bonne connaissance de l'administration des systèmes d'exploitation est nécessaire.

Programme:

Xen Présentation : Introduction aux solutions de virtualisation Xen.
Notion d'hyperviseur.
Les différents types d'hyperviseurs.
La paravirtualisation.
Prérequis matériel pour l'utilisation de Xen.
Systèmes d'exploitation supportés.
Présentation de l'architecture Xen :
l'hyperviseur et les systèmes hôtes.
Les fonctionnalités disponibles :
migration d'un domaine, gestion des ressources, clustering.
Contraintes de sécurité.

Installation : Travaux pratiques d'installation à partir des packages RPMs ou des binaires debian.
Installation de xen et du noyau dom0.
Personnalisation d'un noyau :
compilation du dom0 pour refléter la configuration de la machine hôte.
Configuration et démarrage.
Paramétrage du lanceur : grub.
Création d'un domaine.
Arrêt d'un domaine.

Virtualisation avec Xen

SY004

Domaines utilisateurs	: Gestion des systèmes invités : le service xend La commande d'administration xm (xm create, xm liste...) l'accès à xend par l'interface web. Travaux pratiques : ajout de systèmes invités avec xm create. Utilisation d'un fichier de description de machines virtuelles. Configuration de domU. Arrêt et démarrage de systèmes invités avec xm shutdown et xm reboot.
Supports de stockage	: Déclaration des espaces de stockage accessibles au domU : périphériques blocs, partitions physiques, ou volumes logiques.
Systèmes paravirtualisés	: Etude des paramétres de démarrage des domU. Travaux pratiques : création manuelle d'images. Utilisatoin de debootstrap, et de rpmstrap.
Administration	: Les outils d'administration : Xend, Xm Etude détaillé de la commande xm. Mise en oeuvre de la console d'administration : configuration de domaines, du réseau.
Mise en production	: La gestion des LVM,des processeurs Sauvegarde et restauration de domaines Gestion des ressources : CPu, mémoire, réseau et stockage

Virtualisation avec KVM

SY007

Durée: 2 jours
1190 €

7 au 8 mars
16 au 17 mai
4 au 5 juillet
10 au 11 octobre

Public:
Administrateurs,chefs de projet et toute personne souhaitant mettre en oeuvre la virtualisation avec kvm.

Objectifs:
Comprendre le principe de fonctionnement de kvm, savoir l'installer et l'administrer.

Connaissances préalables nécessaires:
Une bonne connaissance des systèmes d'exploitation est nécessaire.

Programme:

Introduction : Objectifs d'un système d'exploitation, gestion de ressources.Partager des ressources entre plusieurs applications, systèmes...
Notion de virtualisation, quelle granularité (disques, système d'exploitation, machines physiques...)
Historique : VM (Virtual Machine), VMWare, UML, Xen...Les différentes techniques de virtualisation sur Linux.éfinitions : conteneurs, machines virtuelles, hyperviseurs, virtualisation matérielle.
Présentation de kvm : Kernel-based Virtual Machine.Principe et architecture : module intégré dans le noyau Linux, base QEMU.Positionnement par rapport aux autres solutions de virtualisation.Prérequis matériels et logiciels.

Présentation QEMU : Principe de QEMU et architecture.Deux modes de fonctionnement : code compilé pour un processeur,
émulation d'une machine physique.Travaux pratiques : installation et lancement d'une image
Etude des options de lancement de qemu.Consoles des machines virtuelles :graphiques (console VNC, Spice, ..) ou consoles en mode texte.

Virtualisation avec KVM

SY007

Installation de kvm	: Deux configurations possibles :depuis un noyau Linux de version supérieure à 2.6.25 et contenant les modules kvm ou avec recompilation du noyau.Optimisation, gestion de la mémoire.
Travaux pratiques :	: installation avec un noyau contenant les modules kvm Gestion des images :création d'images, différents supports possibles, options de lancement Travaux pratiques :commandes info, check, resize, convert Gestion du matériel:architectures supportées, processeurs, mémoire, périphériques de stockage, audio, video, usb, bluetooth, etc ... Configuration du réseau:différents modes possibles (user, tap, bridge,...) Travaux pratiques :configuration réseau sur les images créées aux chapitres précédents
Migration d'images	: Le besoin.Sauvegarde/chargement de machines virtuelles :à l'arrêt ou en fonctionnement Limites par rapport aux processeurs Snapshots et migrations :principe de fonctionnement Mise en oeuvre et options de la commande 'migrate'. Paramétres (bande passante) Migration vers un fichier : sauvegarde puis restauration
Administration	: Les outils de gestion de machines virtuelles kvm :UVMM, virsh, virt-manager. Travaux pratiques avec libvirt Présentation de proxmox et mise en oeuvre :gestion de machines virtuelles, création de clusters proxmox. Méthode de migration.

Virtualisation avec lxc

SY008

Durée: 2 jours

Prix et dates: nous consulter

Public:
Administrateurs, chefs de projet et toute personne souhaitant mettre en oeuvre la virtualisation avec lxc.

Objectifs:
Comprendre les principes des linux containers et savoir les mettre en oeuvre.

Connaissances préalables nécessaires:
Une bonne connaissance des systèmes Linux est nécessaire.

Programme:

Introduction : Objectifs d'un système d'exploitation, gestion de ressources.
Partager des ressources entre plusieurs applications, systèmes...
Notion de virtualisation, quelle granularité (disques, système d'exploitation, machines physiques...)
Historique : VM (Virtual Machine), VMWare, UML, Xen...
Les différentes techniques de virtualisation possibles sur Linux
conteneurs d'application, noyaux secondaires, machines virtuelles, hyperviseur, virtualisation matérielle...

Présentation de lxc : Linux containers, historique, principe de fonctionnement.
L'isolation de ressources, création d'un environnement utilisateur.
Positionnement par rapport aux autres solutions de virtualisation.

Cgroup : Fonctionnement de Control Group.
Travaux pratiques :Vérification de la configuration du noyau.Activation des Cgroups.

SY008

Virtualisation avec lxc

Les outils LXC	: Site de référence pour le téléchargement.Installation de LXC par rpm, urpmi, yum ou apt-get install. Présentation des différents outils pour :vérifier la configuration du noyau, créer, détruire, gérer les conteneurs, et les tâches associées lxc-checkconfig,lxc-console, lxc-create, lxc-start, lxc-stop etc...Travaux pratiques :vérification de la configuration avec lxc-checkconfig
Gestion des conteneurs	: Configuration, création, démarrage. Travaux pratiques :étude des exemples de configuration dans /share/doc/lxc/examples.Utilisation des templates pour créer des containers standards. Exemple de busybox.Choix des systèmes de fichiers.
Configuration du réseau	: Les différentes méthodes : interface physique, pont/commutateur virtuel, vlan. Exercices pratiques :Mise en oeuvre. Configuration d'un point par brctl.
Exploitation	: Description des ressources à administrer:répertoires du produit lxc fichiers de configuration des containers.Systèmes de fichiers des containers Interventions possibles : visualisation des connexions réseau d'un container, des processus s'exécutant dans un lxc, .. Travaux pratiques :exemples du fichier fstab permettant de configurer les partitions du container Commandes lxc d'arrêt/relance, de vérification des containers et de supervision Exercices avec lxc-console, lxc-monitor, lxc-nestat,lxc-execute, ...
Administration avec libvirt	: Présentation de l'API libvirt et des fonctionnalités apportées,Virtual Machine Manager Travaux pratiques :installation de libvirt et lancement de virt-manager lxc avec libvirt : gestion des containers, création d'images, configuration du réseau, du stockage virtuel, de la mémoire.

Docker : mise en oeuvre

SY009

Durée: 2 jours
1145 € HT

11 au 12 mars
20 au 21 mai
16 au 17 septembre
12 au 13 novembre

Public:
Administrateurs, chefs de projet et toute personne souhaitant mettre en oeuvre Docker pour déployer ses applications.

Objectifs:
Comprendre et savoir mettre en oeuvre Docker, et les produits de l'ecosystème pour déployer des images tout en intégrant les contraintes de production.

Connaissances préalables nécessaires:
Il est demandé aux participants de connaître les bases du système Unix/Linux.

Programme:

Introduction : Présentation docker, principe, fonctionnalités. Besoins : packaging d'applications, déploiements rapides, coexistence de plusieurs versions d'une application sur un même serveur.
Les différentes éditions et leurs fonctionnalités : Docker Enterprise Edition, Docker Community Edition, Docker Cloud

Principe et architecture : Présentation de lxc : Linux containers, historique, principe de fonctionnement. Les Cgroups. L'isolation de ressources, création d'un environnement utilisateur.
Positionnement par rapport aux autres solutions de virtualisation.
Apports de Docker : Docker Engine pour créer et gérer des conteneurs Dockers.
Plate-formes supportées.L'écosystème Docker : Docker Machine, Docker Compose, Kitematic, Docker Swarm, Docker Registry

SY009

Docker : mise en oeuvre

Installation et configuration	: Prérequis techniques.Tavaux pratiques : installation sur Linux.Mise en oeuvre des scripts fournis par Docker pour l'installation. Création d'un groupe Docker.Mise en oeuvre en ligne de commande.Démarrage d'un container simple. Configuration de Docker et des containers.Travaux pratiques :démarrage automatique des containers, contrôle avec systemd, limitation des ressources.
Gestion des images et des conteneurs	: Création de nouvelles images.Principe des DockerFile. Travaux pratiques :Utilisation de DockerFile pour créer des images personnalisées:principales instructions (RUN, FROM, ENV, EXPOSE, etc ...) Recommandations et bonne pratiques d'écriture de DockerFile.Gestion des conteneurs : création, affichage, sauvegarde de l'état Exemple de déploiement d'une application web avec des containers.Présentation du Docker Hub.Publication d'images vers un registry.
Volumes de données	: Initialisation des volumes de données lors de la création d'un container. Travaux pratiques:ajout de volumes de données, contrôle avec la commande Docker inspect;Sauvegarde, migration, restauration de volumes Création de conteneurs de volumes de données
Administration	: Présentation des outils Swarm, Compose, Docker Machine.Fonctionnalités de swarm : cluster Docker, principe du mode swarm, load balancing. Démonstrations de load balancing.Applications de multi-containers avec Compose Méthode d'administration des containers en production.Orchestration avec Docker Machine.Configuration réseau et sécurité dans Docker Présentation des plugins Docker

Docker : administration avancée

SY010

Durée: 2 jours
1250 € HT

13 au 14 mars
22 au 23 mai
18 au 19 septembre
14 au 15 novembre

Public:

Administrateurs, chefs de projet et toute personne souhaitant maîtriser les concepts avancés de Docker

Objectifs:

Savoir configurer les fonctionnalités avancées de Docker :la sécurité, les configurations multi-hôtes, la création de registres privés, le provisionning de services dans le cloud, ...

Connaissances préalables nécessaires:

Il est demandé aux participants de connaître les bases du système Unix/Linux et les bases de Docker , ou d'avoir suivi le stage "Docker : mise en oeuvre".

Programme:

Docker engine : Fonctionnalités,installation et configuration

Le service Docker : Docker daemon : rôle, configuration des principales options.
Option socket pour les accès en réseau.
Variables d'environnement : DOCKER_HOST, et DOCKER_TLS_VERIFY
Option storage-driver :
définition des formats de stockage des images.
Gestion de noeuds avec l'option -cluster-advertise
Travaux pratiques :
configuration des accès réseau et de clusters Docker

Création d'un registry privé : Présentation de Docker Trusted Registry (DTR).
Architecture. Containers et volumes propres au DTR
Pilotage par UCP (Universal Control Plane).
Travaux pratiques :
installation d'un dépôt privé.
Gestion des images du DTR, des droits d'accès.

SY010

Docker : administration avancée

Administration en production : Applications multi-containers avec Compose:
définition de l'environnement applicatif,
déclaration des services dans docker-compose.yml,
exécution avec docker-compose.
Méthodes d'administration de containers en production.
Orchestration avec Docker Machine.
Travaux pratiques :
exemples de provisionning en environnement mixte,
dans le cloud et sur des machines physiques.
Présentation de Swarm pour le clustering :
fonctionnalités, gestion de clusters docker,équilibrage de charge,
répartition de tâches, gestion de services répartis,..

Sécurité : Analyse des points à risques :
le noyau, le service Docker, les containers, ..
et des types de dangers : déni de service, accès réseau non autorisés, ..
Mécanismes de protection :
pile réseau propre à chaque container,
limitations de ressources par les cgroups,
restrictions des droits d'accès sur les sockets,
politique de sécurité des containers.
Travaux pratiques :
mise en évidence de failles de sécurité et des bonnes pratiques à adopter.
Sécurisation des clients par des certificats
Principe, et mise en oeuvre avec openssl.
Fiabilité des images déployées dans Docker:
présentation de Content Trust pour signer les images.
Exercices pratiques :
activation de Content Trust,
variable d'environnement DOCKER_CONTENT_TRUST,
Création et déploiement d'images signées.
Configuration réseau, sécurité et TLS

Réseaux virtuels avec OpenvSwitch

SY012

Durée: 2 jours
1215 € HT

18 au 19 mars
6 au 7 juin
17 au 18 octobre

Public:
Administrateurs responsables d'un parc de machines virtuelles.

Objectifs:
Connaître les techniques et outils de configuration et administration de réseaux virtuels.

Connaissances préalables nécessaires:
Connaissance de l'administration système et des réseaux.

Programme:

Introduction	: Le besoin de réseaux virtuels : multiples machines virtuelles sur un seul hôte, cloud Outils et leurs caractéristiques : VDE, Switch Light, OpenVswitch
Réseaux virtuels et clouds	: Exemples dans OpenStack, et Amazon AWS. Création d'interface réseaux, configuration du routage, mise en place des services réseaux (DNS, DHCP, ...)
Open vSwitch	: Présentation : fonctionnalités (commutateur virtuel, sécurité, QoS, ..) architecture, protocoles supportés. Installation : packages et partie noyau. Configuration par ovs-vsctl.
Administration réseaux virtuels	: Travaux pratiques avec Open vSwitch Création des interfaces réseaux. Activation des interfaces et du routage. Supervision réseaux, analyse des flux. Configuration de la QoS pour limiter les flux.

Kubernetes : optimisation des conteneurs

SY180

Durée: 2 jours
1250 € HT

4 au 5 mars
13 au 14 mai
23 au 24 septembre
2 au 3 décembre

Public:
Administrateurs, chefs de projet et toute personne souhaitant mettre en oeuvre kubernetes pour le déploiement d'applications

Objectifs:
Comprendre le fonctionnement de kubernetes, savoir l'installer, le configurer et l'administrer

Connaissances préalables nécessaires:
maitrise des systèmes Linux, des réseaux tcpip, et des concepts de virtualisation et containers

Programme:

Introduction : Présentation Kubernetes, origine du projet,
Fonctionnalités: automatisation des déploiements et de la maintenance des applications en containers.
Containers supportés, plate-formes utilisant Kubernetes.
Composants de Kubernetes.
Définitions: pods, labels, controllers, services

Architecture : Kubernetes Master: stockage des configurations par etcd, interfaçage par l'API server,
noeuds Kubernetes: hébergement des containers,
Kubelet pour la supervision des noeuds.

Installation et configuration : Présentation des différentes solutions d'installation
Installation des outils : kubectl,minikube,kubeadm
Configuration de pods et containers:
assignation de mémoire, espace de stockage, processeurs,
affectation de pods à des noeuds.
Configuration d'applications et exécution.

SY180

Kubernetes : optimisation des conteneurs

Administration	: Outils de supervision, analyse des logs, debugging Utilisation de kubectl exec pour accéder en shell à un container Analyse de l'état des noeuds avec Node Problem Detector Mise en oeuvre de StackDriver
Sécurité	: Présentation des points à sécuriser Accès à l'API Kubernetes Limitations des ressources Contrôle des accès réseau Restrictions des accès à etcd

Cloud : technologies et enjeux

SY100

Durée: 1 jour
670 € HT

25 février
11 juin
30 septembre

Public:
Architecte, chef de projet, et toute personne souhaitant comprendre la notion de cloud, et plus précisément les solutions de cloud d'entreprise.

Objectifs:
Connaître les possibilités des solutions de cloud, ainsi que les contraintes de mise en oeuvre.

Connaissances préalables nécessaires:
Connaissance générale des systèmes d'informations.

Programme:

Introduction	: Définition : cloud computing, Les types de cloud: Saas, Paas, Iaas. Clouds privés et publics. Fonctionnalités disponibles : partage de données, stockage distant (services EBS et S3 d'Amazon) mise à disposition de services (SNS, SQS, ...), applications, pilotage distant de systèmes locaux, mise à disposition de ressources à la demande.
Technologies	: virtualisation, services Web Les acteurs du marché des clouds : Amazon, Eucalyptus, IBM, microsoft, openStack, openNebula, cloudStack, rackspace, salesForce. Comparaison des offres.
Positionnement par rapport aux autres architectures	: Positionnement par rapport aux centres de données (datacenter) : vSphere La gestion de serveurs IBM en grappe : Capacity On Demand, console HMC et micro-partitionnement. Les architectures JEE en grappes. Présentation de WebSphere Cloud. Comment migrer les applications vers un cloud ?

SY100

Cloud : technologies et enjeux

Cloud d'entreprise	: Comment s'organise une infrastructure d'entreprise en cloud ? Comment migrer les services reseau : dns, dhcp, smtp, ... Organisation des machines virtuelles en sous-réseaux, switches et cables virtuels. Découpage d'une grappe cloud en zones. Urbanisation. Les aspects sécurité : évaluation des risques, présentation des solutions. Les répartiteurs de charges disponibles dans les clouds. Est ce que le cloud sera compatible IPv6 ? Comment intégrer la voix sur IP dans un cloud ?

SY101

Architecture cloud d'entreprise

Durée: 3 jours
1760 €

26 au 28 février
12 au 14 juin 1er au 3 octobre

Public:

Architecte, chef de projet, et toute personne souhaitant comprendre la notion de cloud, et plus précisément les solutions de cloud d'entreprise.

Objectifs:

Comprendre les possibilités des solutions Cloud, ainsi que les contraintes de mise en oeuvre. Savoir exploiter les nouvelles fonctionnalités de gestion d'une infrastructure et de ressources distribuées et l'industrialisation induite par les solutions. Cette formation s'appuie sur de nombreux exemples de mise en oeuvre.

Connaissances préalables nécessaires:

Connaissance générale des systèmes d'informations.

Programme:

Introduction : Définition : cloud computing, mise à disposition de ressources selon les besoins. Elasticité.
positionnement par rapport aux autres architectures. Notion de aaS : IaaS, Paas, Saas.
Architecture générale d'un cloud. Besoins techniques. Aspects juridiques et sécurité.
Exemples : AWS, Google, Rackspace, Azure
Présentation rapide de ressources Cloud avec AWS
Déploiement d'une application en mode Saas

Construction d'un cloud : Principes. Architecture. Technologies utilisées.
Outils de traitements et d'administration en environnement distribué;
gestion de datacenter, virtualisation de serveurs, conteneurs, virtualisation de réseaux

SY101

Architecture cloud d'entreprise

Mise en oeuvre	: Travaux pratiques avec OpenStack. Présentation. Architecture des produits. Travaux pratiques : installation d'un cloud interne sur plusieurs machines physiques. Administration avec la console Web et en ligne de commandes (gestion des noeuds, des utilisateurs, ...) Lancement d'une première machine virtuelle. Ajout de noeuds physiques. Mise en évidence de l'élasticité.
Architecture	: Zoom sur la partie réseau Mise en oeuvre du réseau entre machines virtuelles. SDN et NFV. OpenVSwich. Zoom sur les ressources de calculs. Introduction aux micro-services. Choix : VM, conteneurs LXC, conteneurs Docker/Rocket.
Intégration	: Projets OpenStack. Gestion des conteneurs avec Magnum. Lancement d'un conteneur. Accès aux fermes BigData Hadoop. Le projet Sahara.
Interopérabilité	: Mise en oeuvre d'un cloud avec openNebula. Mise en évidence de l'interopérabilité avec Amazon AWS et OpenStack Transferts de ressources entre les différents cloud.
La sécurité	: Evaluation des risques, présentation des outils et solutions Aspects juridiques, protection des données, traçabilité, chiffrement, génération de clés... Protection des accès.
Disponibilité	: Les points de faiblesse d'un cloud. Passage en mode HA. Sécurisation des données par distribution et réplication. Introduction à Ceph. Installation d'un serveur S3 local. Sécurisation des données avec Swift. Mise en mode stateless des noeuds physiques de calculs. Migration à chaud des VM. Sécurisation des tables de méta-données.

OpenStack : configuration et administration

SY111

Durée: 3 jours
1840 €

4 au 6 février — 7 au 9 octobre
24 au 26 avril — 9 au 11 décembre

Public:
Architecte, chef de projet, et toute personne souhaitant installer une infrastructure de cloud avec OpenStack

Objectifs:
Comprendre le fonctionnement d'OpenStack, savoir le déployer, le configurer et l'automatiser.Savoir gérer des machines virtuelles avec OpenStack, configurer le stockage virtuel et les réseaux viruels.

Connaissances préalables nécessaires:
Connaissance générale des systèmes d'informations, systèmes et réseaux IP.

Programme:

Introduction : Présentation d'OpenStack : historique, acteurs, licence
Fonctionnalités :
outils d'orchestration de cloud,
stockage élastique, gestion d'images virtuelles, réseaux virtuels,
compatibilité Amazon EC2, EBS, S3, ...
Possibilité de créer des clouds privés ou des clouds hybrides avec AWS.

Caractéristiques techniques : Prérequis : plate-formes Linux,
Hyperviseurs et systèmes de stockage supportés.
Architecture d'OpenStack : notion de services et de projets
Présentation des principaux projets, et de leurs rôles :
Nova, Cinder, Glance, Swift, Neutron, Keystone, Horizon
Principe d'interrogation des services : accès par des clients webservices.
Etude de l'architecture réseau d'OpenStack :
cloud controller, noeuds avec stockage et exécution d'instances virtuelles, serveur de monitoring
Outils et interfaces de gstion :
le Dashboard, l'interface de gestion, pilotage, suivi.

SY111

OpenStack : configuration et administration

Installation et configuration : Prérequis matériel et logiciel.
Etude des phases de l'installation et des composants à installer :
authentification avec Keystone, serveur d'images Glance,
stockage d'objets avec Swift,
serveur de calcul Nova,
services auxiliaires : dashboard, FlatDHCP, MySQL ou postgreSQL
Présentation de packstack.
Définition des tenants.
Préparation de l'installation.
Travaux pratiques :
installation à partir de scripts,
identification des différents types de machines virtuelles dipsonibles,
vérification de l'état des services,
configuration de la base de données, du service de messages,
du service keystone : gestion des utilisateurs, groupes, accès;
configuration du réseau,
création et lancement des instances.
Déploiement en masse. Scripts d'automatisation.

Utilisation de la console Web : Présentation des fonctionnalités offertes par la console.
Notions de projets.
Travaux pratiques :
création de nouveaux projets, d'utilisateurs,
génération des clés pour la connexion aux instances,
création d'images et lancement des instances,
test de différents moyens d'accès : par une console VNC, par ssh

Extension : Mise en pratique :
Ajout d'un noeud de calcul. Configuration du nouveau noeud.
Visualisation de la capacité supplémentaire.
Migration d'un noeud de calcul.
Passage à l'échelle.
Les mises à jour.
Analyse des problèmes.

OpenStack : configuration et administration

SY111

Gestion des volumes	: Présentation de Cinder. Architecture, locale, NAS, SAN. Travaux pratiques: Mise en oeuvre : démarrage du service, installation de volumes iScsi. Manipulation de disques virtuels : création, attachement, formatage, suppression. Analyse de stratégies pour le stockage.
Gestion du réseau	: Principe : virtualisation des éléments d'un réseau physique : ponts, commutateurs, adressage, routage, répartition de charge, pare-feux, .. Définitions et terminologie : adresses mobiles, adresses fixes, Virtual network, physical network, flat network, provider network. Les modes réseaux : Flat, FlatDHCP, VLAN. Présentation de Neutron : fonctionnalités, architecture. Travaux pratiques : Mise en oeuvre de Neutron. Création d'un réseau privé virtuel. Ajout de routeurs virtuels et liaisons avec les instances. Création de stocks d'adresses IP mobiles. Affectation des adresses. Scripts SDN (Software Defined Network). Définition d'une architecture globale.
Les utilitaires EC2 Tools	: Principe de fonctionnement des EC2 tools. Travaux pratiques : téléchargement des identifiants de connexion, gestion et installation d'images, lancement de nouvelles instances, .. Utilisation de la compatibilité Amazon EC2
Exploitation d'une infrastructure OpenStack en production	: Présentation des fonctionnalités disponibles en ligne de commande avec Nova. Travaux pratiques : gestion de services, des journaux, ajouts de noeuds, configuration, migration de machines virtuelles d'un noeud de calcul vers un autre

Cloud d'entreprise avec OpenNebula

SY113

Durée: 3 jours
1840 €

18 au 20 février
15 au 17 avril
1er au 3 juillet
14 au 16 octobre

Public:

Architecte, chef de projet, et toute personne souhaitant installer une infrastructure de cloud avec OpenNebula

Objectifs:

Savoir installer OpenNebula, le configurer et l'utilisation pour le déploiement de machines virtuelles. Tous les concepts abordés dans cette formation sont illustrés dans de nombreux travaux pratiques.

Connaissances préalables nécessaires:

Connaissance de l'administration des systèmes Linux et réseaux IP.

Programme:

Introduction : Fonctionnalités : gestion de machines virtuelles, d'images, de réseaux virtuels et de stockage
Historique projet OpenNebula, écosystème, support OpenNebula Systems.

Caractéristiques techniques : Hyperviseurs supportés : xen, kvm, VMware. Notion d'instance OpenNebula et de VDC 'Virtual Data Centers'. Fédérations.
Compatibilité EC2. Cloud-bursting.
Présentation des différentes APIs disponibles : Ruby, Java, XMLRPC
Architecture OpenNebula : management daemon et scheduler
Haute disponibilité et outils supervision.

Cloud d'entreprise avec OpenNebula

SY113

Installation et configuration	: Prérequis matériel et logiciel. Installation depuis les packages et démarrage : partie frontend, et noeuds. Gestion des utilisateurs et accès de base. Interface d'administration : les différentes méthodes : ligne de commande ou interface Sunstone. Sécurité : gestion des utilisateurs, groupes et droits d'accès aux ressources. Lien vers ldap et Active Directory.
Stockage	: Différents types de stockage : filesystem, LVM, Ceph et vmdk. Manipulation de disques virtuels : création, attachement, formatage, suppression. Stratégie pour le stockage.Gestion des images virtuelles.
Gestion des machines virtuelles	: Hyperviseurs. Gestion de modèles. Gestion des machines : création, cycle de vie, migration. Modifications des caractéristiques. Elasticité. Groupement de machines virtuelles. Notion de service. Mise en oeuvre de OneFlow. Suivi des consommations.
Gestion du réseau	: Notion de réseau de service. Réseaux virtuels, routeurs virtuels. Mise en oeuvre. Différents types de drivers associés à chaque hôte Intégration avec Ebtables, VLan, VxLan, OpenvSwitch

Cobbler : gestion de configurations

SY123

Durée: 2 jours

Prix et dates: nous consulter

Public:

Administrateurs, exploitants souhaitant utiliser Cobbler pour la gestion centralisée des configurations logicielles.

Objectifs:

Comprendre le fonctionnement de Cobbler, et savoir le mettre en oeuvre pour une administration centralisée.

Connaissances préalables nécessaires:

Il est demandé aux participants de connaître l'administration des systèmes Linux et un langage de développement de scripts.

Programme:

Introduction	: Présentation de Cobbler, fonctionnalités. Gestion automatisée en réseau des installations de systèmes Linux : serveurs, machines virtuelles,containers. License, version et utilisations classiques de cobbler dans le monde Linux.
PXE et kickstart	: Principe de fonctionnement de PXE et étude de l'installation automatisée de Linux CentOS avec kickstart.
Installation cobbler	: Installation avec yum et configuration dans le fichier /etc/cobbler/settings. Démarrage du service cobbler. Premiers déploiements simples : import d'une distribution et création d'un système à déployer.Présentation de l'interface client de cobbler.
Les primitives	: Modélisation d'une distribution avec Cobbler. Les objets, les règles standards, l'héritage.
Les commandes	: Les commandes d'interrogation des objets : list, report, remove, copy, find, etc ... Administration : Check, sync, import, buil. Recherche en ligne de commande : dsitro, profile, system, repo , 'find' Présentation de l'interface web.

Ansible : industrialiser les déploiements

SY124

Durée: 2 jours
1290 €

7 et 8 février
18 et 19 avril
6 et 7 juin
26 et 27 septembre
7 et 8 novembre
16 et 17 décembre

Public:
Administrateurs, exploitants souhaitant mettre en oeuvre Ansible pour le déploiement des services et applications.

Objectifs:
Comprendre le fonctionnement d'Ansible, savoir rédiger des scripts de déploiement

Connaissances préalables nécessaires:
Il est demandé aux participants de connaître l'administration des systèmes Linux et un langage de développement de scripts.

Programme:

Introduction : Gestion automatisée de l'infrastructure systèmes et réseaux (serveurs, machines virtuelles,containers, équipements réseaux)
Notion de playbooks.
Langage de configuration, déploiement, orchestration
Commandes Ad-Hoc

Installation et inventaire avec Ansible : Travaux pratiques sur CentOS.
Configuration de la connexion avec les serveurs distants : ssh
Premiers pas avec ansible :
inventaire des serveurs accessibles : hôtes, groupes, etc ...
Configuration d'un inventaire automatique.

playbooks et modules : Exemples de playbooks
Les modules fournis avec Ansible, écriture de nouveaux modules

commandes Ad Hoc : commandes shell, gestion du parallèlisme,
transfert de fichiers,
gestion des utilisateurs et groupes,
déploiement à partir des sources,
administration des services.

Puppet : administration centralisée

UX161

Durée: 3 jours
1660 €

25 au 27 février
11 au 13 juin
7 au 9 octobre
9 au 11 décembre

Public:
Administrateurs, exploitants souhaitant utiliser Puppet pour la gestion centralisée des configurations logicielles de multiples serveurs ou machines virtuelles.

Objectifs:
Comprendre le fonctionnement de Puppet, et savoir le mettre en oeuvre pour une administration centralisée des configurations.Ce stage est illustré par de nombreux travaux pratiques sur le développement des manifests puppet.

Connaissances préalables nécessaires:
Il est demandé aux participants de connaître les bases du système Unix.

Programme:

Introduction : Présentation du besoin et des fonctionnalités de Puppet.
Gestion des configurations logicielles.
Collecte et centralisation des informations.
Définition des configurations cibles, mises à jour automatiques ou manuelles.
L'orchestration dans le cas d'un cloud d'entreprise.
La gestion de configuration de machines virtuelles.
Historique de puppet et les différentes versions de puppet.
Présentation de Puppet Enterprise.

Architecture : Principe client-serveur, modules de configuration, les agents Puppet, la console,
l'outil de gestion de cloud, les manifests puppet.
Présentation des plate-formes supportées comme serveur et comme client (agent puppet).
Mode opératoire : définition des configurations,
vérification de l'état des clients, simulation des changements proposés par Puppet,application sur les systèmes cibles.

Puppet : administration centralisée

UX161

Installation et configuration	: Prérequis systèmes. Travaux pratiques :Installation d'une infrastructure Puppet :serveur, base de données, agents. Etude du fichier de configuration puppet.conf. Configuration des rôles : maître, agent, console...Gestion des certificats sur les clients.
Le langage puppet	: Introduction : présentation des manifests, modules, templates, des modules disponibles sur la forge puppet. Les manifests : description du langage déclaratif de puppet. Syntaxe, dépendance entre instructions. Etudes des variables, présentation des facts.Déclarations conditionnelles : if, case, selectors Travaux pratiques :rédaction d'un exemple de manifest de création d'un fichier et modification des droits, application de ce manifest sur un poste client cible. Etude détaillée des 'ressources' puppet :file, package, service, user, exec, notify, .. Exemple de la ressource 'file' :différents attributs disponibles pour déterminer les caractéristiques d'un fichier, s'assurer de sa présence, ... Exemple de la gestion de dépendance :production de manifest pour l'installation de sshd Les classes et modules. Installation de modules. Travaux pratiques :création d'un module pour propager la configuration de bash sur les postes clients. Définition des templates.Exemple de création de templates en utilisant le langage ERB. Etude des paramètres de classes.Définition de nouveaux types de ressources. Mise en oeuvre sur de nombreux travaux pratiques. Utilisation et génération de documentations au format puppet. Organisation du site .pp.
Bonnes pratiques	: Retours d'expériences et méthodes d'organisation et de développement des scripts puppet.Présentation des patterns pour puppet.

Puppet : expertise

UX162

Durée: 2 jours

Prix et dates: nous consulter

Public:
Administrateurs, exploitants souhaitant approfondir leurs connaissances de Puppet pour la gestion centralisée des configurations logicielles.

Objectifs:
Comprendre le fonctionnement le fonctionnement avancé de puppet, de la base hiera, savoir rechercher et créer des modules et connaître les outils complémentaires disponibles autour de puppet.

Connaissances préalables nécessaires:
Il est indispensable de connaître les bases de puppet.

Programme:

La base hiera : Intérêt de Hiera :
gestion des paramètres en dehors des manifests.
Fonctionnement, mise en oeuvre .
Travaux pratiques :
configuration avec le fichier hiera.yaml
Préparation de la hiérarchie
Utilisation des données hiera depuis puppet.

Les outils complémentaires : Facter : pour le recensement des informations des clients.
Ralsh : un langage de script shell.
Puppet Dashboard : interface de rapport d'activité des agents.
Mcollective : pour exécuter des commandes en parallèle sur les serveurs cibles.
La forge de modules et Puppet Module Tool pour utiliser des modèles de configurations.

UX162

Puppet : expertise

Les modules : Présentation de "Puppet Forge" et recherche de modules.
Etude de modules disponibles sur 'puppet forge' :
puppetlabs/stdlib, utilisé dans la plupart des autres modules,
puppetlabs/ntp : pour synchroniser les postes ou machines virtuelles,
puppetlabs/apt, puppetlabs/firewall, ...
Le langage de description de configuration.
Travaux pratiques:
Création de modules, mise en oeuvre, et dépôt sur la forge.
Exemple : lien avec Nagios par le module thias/nagios

Cloud Provisionner Puppet : Présentation : gestion des instances de machines virtuelles dans le cloud
Travaux pratiques :
installation de Cloud Provisioner et démarrage d'instances
Exemple d'utilisation :
cas du bootstrap pour installer puppet dès la création de la machine virtuelle.

Traitement des rapports : Description du fonctionnement et des différentes étapes :
activation dans le ficher puppetconf,
stockage par le puppet Master.
Travaux pratiques :
installation de la Dashboard et configuration de l'environnement,
création d'un nouvel utilisateur,
préparation de la base,
activation des jobs des delayed jobs,
gestion des rapports émis par les agents puppet

MCollective : Présentation des fonctionnalités
Principe de fonctionnement Middleware
Installation et premier test : validation des clients collectés.

Gestion de configuration avec Chef

UX170

Durée: 3 jours
1660 € HT

6 au 8 février
23 au 25 mai
11 au 13 septembre
27 au 29 novembre

Public:
Administrateurs, exploitants souhaitant utiliser Chef pour la gestion centralisée des configurations logicielles.

Objectifs:
Comprendre le fonctionnement de Chef, et savoir le mettre en oeuvre pour une administration centralisée.

Connaissances préalables nécessaires:
Il est demandé aux participants de connaître les bases du système Unix/Linux et un langage de développement de scripts.

Programme:

Introduction : Présentation de Chef, fonctionnalités
Gestion automatisée de l'infrastructure systèmes et réseaux (serveurs, machines virtuelles,containers, équipements réseaux)

Architecture : Chef server, Chef Analytics pour le suivi,
Chef management Console : interface web d'administration
Chef-client sur les noeuds
clients d'administration (workstation) et Chef Development Kit.
Définition de la notion de noeud, et des attributs du noeud, des cookbooks

Installation et configuration : Chef Server :prérequis techniques,
différents modes d'installation (standalone, cluster, ..)
Clients d'administration (workstation):
utilisation de knife pour synchroniser les données avec chef-server.
Installation de chef-client sur un noeud,
Etapes de l'exécution d'un client :
Récupération des données sur les noeuds, authentification auprès du Chef-Server
Création de la "run-list", exécution, mise à jour du noeud.

Gestion de configuration avec Chef

UX170

Cookbooks	: Principe. Les cookbooks disponibles en opensource Exemples : apache2 et nginx pour configurer un serveur apache et nginx, chef-client, pour gérer le fichier de configuration clien.rb et chef-client service Modification, développement de cookbook.
Chef Analytics et la console	: Installation et configuration. Principe de Chef Analytics. Collecte de données, sur les noeuds, les actions exécutées,... Visualisation des données Console web d'administration des clients, cookbooks, noeuds, rapports, rôles, etc..

AWS : les fondamentaux

SY200

Durée: 1 jour
660 €

25 janvier — 5 juillet
23 avril — 12 novembre

Public:
Architecte, chef de projet, et toute personne souhaitant connaître les possibilités du cloud Amazon.

Objectifs:
concevoir une architecture applicative avec AWS.

Connaissances préalables nécessaires:
Connaissance générale des systèmes d'informations et de la virtualisation.

Programme:

Introduction : Rappels sur les définitions du cloud.Présentation du système AWS.
Positionnement par rapport aux autres offres de cloud (GCP,Azure, OpenNebula,openStack, ...)
Etude des fonctionnalités accessibles avec AWS Management Console :
Ressources de calcul et réseaux.Stockage.Bases de données.Déploiement et supervision.
Services applicatifs, services pour mobiles, objets connectés, ...Principe de la compatibilité EC2 et S3. APIS. Interopérabilité, automatisation.
TP : mise en oeuvre de scripts et programme clients AWS

Stockage : Présentation des différentes options de stockage : RedShift, S3, dynamoDB.Mise en oeuvre de Amazon Simple Storage Service (S3)

Calcul et réseaux : Utilisation d'Amazon EC2 (Elastic Compute Cloud) Création de VM. Les AMIs disponibles. MarketPlace.
Les gabarits disponibles. Les droits d'accès, gestion des clés.Paramètres des machines : Elastic Block Storage (EBS), adresses IP élastiques,

Sécurité : Authentification et autorisation dans le cloud.Présentation AWS Identity et Access Management.

AWS: développement

SY201

Durée: 3 jours
1840 €

18 au 20 février
1er au 3 mai
11 au 13 septembre
13 au 15 novembre

Public:

Chefs de projet, architectes, développeurs, et toute personne souhaitant connaitre les outils et technologies pour concevoir une application dans l'environnement AWS.

Objectifs:

Savoir quels sont les services disponibles dans l'environnement AWS pour élaborer une architecture applicative performante, ains que les outils à destination des développeurs.

Connaissances préalables nécessaires:

Connaissances des fondamentaux du cloud AWS et des bonnes pratiques de développeurs

Programme:

Services applicatifs	: Rappels des services de base: EC2 (Elastic Compute Cloud), VPC (Virtual Private Cloud), S3 (Simple Storage Service), EBS (Elastic Block Storage), RDS (Relational Database Service) Présentation des services applicatifs :calculs distribués avec EMR (Elastic MapReduce) et Redshift gestion de contenus avec CloudFront,messagerie et workflow avec SNS (Simple Notification Service) et SES (Simple Email Service)
Interfaces de gestion des services	: AWS Management Console:navigation dans les différents services et de l'utilisation de la Console. Automatisation, scripts de gestion des services avec AWS CLI Travaux pratiques : création d'une architecture applicative simple
Services SNS et SES	: Etude et mise en oeuvre des services de notifications et messagerie

SY201

AWS: développement

Outils de développement	: AWSCode, contrôle des sources avec AWS CodeCommit, intégration et diffusion continue avec CodePipeline automatisation des déploiements avec CodeDeploy Présentation des kits SDK disponibles sur AWS

AWS : opérations système

SY202

Durée: 3 jours
1840 €

11 au 13 mars
17 au 19 juin
14 au 16 octobre

Public:
Administrateurs, architectes souhaitant savoir quels sont les outils et méthodes pour effectuer les opérations systèmes dans un cloud amazon.

Objectifs:
Connaître les outils d'exploitation et d'administration disponibles dans le cloud AWS.

Connaissances préalables nécessaires:
Connaissances des fondamentaux du cloud AWS.

Programme:

Services d'administration	: Rappels des services de base: EC2 (Elastic Compute Cloud), VPC (Virtual Private Cloud), S3 (Simple Storage Service), EBS (Elastic Block Storage), RDS (Relational Database Service) Présentation des services d'administration : gestion de la sécurité, supervision, déploiement d'applications, automatisation
Interface de gestion des services	: AWS Management Console:navigation dans les différents services et de l'utilisation de la Console. Automatisation, scripts de gestion des services avec AWS CLI
Calculs	: Paramètrage de machines EC2. Comparaison avec les conteneurs. Présentation des services ECS, docker et EKS, kubernetes. TP : mise en oeuvre d'un conteneur et gestion de son cycle de vie.
Réseaux	: VPC : Mise en place d'un réseau privé virtuel. Regroupement des VMs dans un même VPC. Déclaration d'adresses IP élastiques. Attachement aux VMs. Route53 : déclaration d'un nom de domaine et attachement à une adresse IP

SY202

AWS : opérations système

Stockage	: Automatisation d'envois vers S3. Mise en oeuvre d'un minio sur une ferme locale et interconnexion avec aws. EXposition du stockage AWS en local : mise en oeuvre du service Storage Gateway Archivage de données avec Glacier.
	:
Sécurité	: Gestion des accès.Utilisation de la console IAM, création d'utilisateurs et groupes. Les rôles prédéfinis. Création d'un rôle.Ajout d'utilisateurs, de groupes. Affectation aux rôles. Mise en oeuvre sur les ressources AWS.Gestion de certificats : création, utilisation. Gestion des firewalls. TP mise en oeuvre WAF
Supervision infrastructure AWS	: Présentation d'Amazon CloudWatch.Mise en oeuvre des scripts CloudWatch Exemples: supervision utilisation mémoire, charge CPU, disques Analyse des logs avec CloudWatch Logs

AWS : BigData avec Hadoop EMR

SY203

Durée: 3 jours
1840 €

13 au 15 février
8 au 10 avril
18 au 20 novembre

Public:
Architectes, chefs de projets souhaitant bénéficier des services offerts par la distribution Hadoop fournie par AWS

Objectifs:
Savoir mettre en oeuvre les techniques de calcul distribué avec Hadoop EMR

Connaissances préalables nécessaires:
Connaissance des fondamentaux du cloud

Programme:

Présentation AWS Hadoop EMR : Rappels des services de base:
EC2 (Elastic Compute Cloud), VPC (Virtual Private Cloud), S3 (Simple Storage Service), EBS (Elastic Block Storage), RDS (Relational Database Service)
Caractéristiques du calcul distribué et du service EMR (Amazon Elactis MapReduce)

La distribution Hadoop EMR : Historique du projet hadoop
Les fonctionnalités : stockage, outils d'extraction, de conversion, ETL, analyse, ...
Exemples de cas d'utilisation sur des grands projets.
Les principaux composants :HDFS pour le stockage et YARN pour les calculs.
Les distributions et leurs caractéristiques. Composants de la distribution Hadoop EMR

Mise en oeuvre : Démonstrations sur une architecture Hadoop multi-noeuds.
TP : mise en place d'une configuration de base avec HBase. Cycle de fonctionnement. Transferts par s3. Envoi des travaux. Visualisation des résultats.
Suivi des travaux avec Hue. Interactivité avec les Notebooks : jupyter.
Configuration des composants de la distribution.

SY203

AWS : BigData avec Hadoop EMR

Exploitation : Gestion des évènements avec Events. Contrôle du réseau et des VPC.
Automatisation de l'exécution sur AWS depuis un poste local. Suivi distant des travaux.
Journalisation, visualisation des logs.Utilisation de la ferme aws en débordement d'une ferme locale.
Sécurité : mise en place d'une configuration de sécurité. Liens avec IAM.

Optimisation : Analyse des performances.Déploiement d'une configuration avec Spark. Evaluation des performances par rapport à une ferme locale. Comparaison des coûts.
Mise en place d'automates d'optimisation.

Filières Réseaux et TCP/IP

Introduction aux réseaux
TCP/IP : protocoles et mise en oeuvre

Migration vers IPv6
Messagerie
Mise en oeuvre SNMP
Annuaire ldap

Introduction aux réseaux

RS001

Durée: 4 jours
2200 €

28 au 31 janvier
1er au 4 avril
1er au 4 juillet
14 au 17 octobre

Public:

Toute personne souhaitant acquérir des connaissances générales sur les réseaux, et plus particulièrement sur la mise en oeuvre d'un réseau et les outils nécessaires à son exploitation.

Objectifs:

Comprendre les composants fonctionnels d'un réseau informatique. Analyser les possibilités d'interconnexion entre les différents réseaux. Connaître l'état de l'art de la conception, de la gestion et du suivi de réseaux hétérogènes.

Connaissances préalables nécessaires:

Aucune connaissance préalable n'est requise pour suivre ce cours

Programme:

Introduction : Le besoin de communication
Quelques définitions. Les couches ISO.
Normalisation et standards (ISO, IEEE, IETF, ATM Forum, ...)

Architectures de base : Topologies filaires, topologies sans fils. Réseau maillé. Doublement de lignes, sécurisation

Supports physiques : Evolutions technologiques et mutation des réseaux.
Acteurs du marché: opérateurs, fournisseurs, intégrateurs, distributeurs.
Câblage : topologies et architectures. Usage des locaux techniques. Brassage.
Radio : le besoin, les limites, l'état du marché.

Transmissions : Pourquoi et comment transmettre les informations ? Des transmissions série, parallèle ou hertzienne aux protocoles.
Concepts de base et terminologie. Composants des réseaux (produits CISCO, 3COM, ...).

Introduction aux réseaux

RS001

Technologies	: Présentation rapide Ethernet, Giga Ethernet, Token-Ring, FDDI, Frame Relay, RNIS, ATM Les Ethernet : du 10M au 10G. Les normes 802.3ab et 802.3ae
Réseaux sans fils Wlan	: HiperLAN. IEEE 802.11
Normes Wifi	: Présentation. Points forts, points faibles. Architecture des réseaux Wifi : 802.11, exemple d'ESS, le monde ad hoc, OLSR Le matériel, interopérabilité
Utilisation du Wifi	: Points d'accès. Modes de fonctionnement, mode répéteur, Mode pont Alignement d'antennes, supervision de réseaux
TCP/IP	: Définitions, adressage. Exemple d'application Le protocole IP, la trame IP, TCP, UDP
Outils réseau	: Outils de trace, tcpdump, outils de diagnostic actifs/passifs, analyseurs de flux, ...
Interconnexion de réseau et routage	: Technologies, commutation. Routage IP. Fragmentation , VLAN. Outils de gestion du routage. Plan d'adressage. QoS.
IPV6	: Besoin, fonctionnalités. La trame IPV6, adressage.
Sécurisation	: VPN et tunnels: Objectif, fonctionnement DMZ et Pare-feux : Définition, serveur Proxy, fonctionnement pare-feux et tunneling Filtrage: les iptables, politique par défaut, etat des connexions, traduction d'adresses, traduction de ports, connexion à internet

RS001

Introduction aux réseaux

Voix sur IP : Commutation de paquets. Avantages de la voix sur IP
Les protocoles : H323, SIP. Introduction RTP : définition et applications, RTP et Nat
Utilisation du registar SIP avec Asterisk. Création des comptes téléphones, du dialplan, verification et tests
Enregistrements SRV : serveurs DNS et Asterisk
Transport de données
Bande passante et qualité de service (QoS)

Evolutions : L'adressage IP, la sécurité, les réseaux de stockage.

TCP/IP : protocoles et mise en oeuvre

RS003

Durée: 4 jours
1930 €

11 au 14 février — 7 au 10 octobre
20 au 23 mai — 2 au 5 décembre

Public:

Toute personne souhaitant mettre en oeuvre TCP/IP et les outils nécessaires à son exploitation.

Objectifs:

Maîtrisez les fonctionnalités du protocole TCP/IP, sa position par rapport aux autres protocoles. Savoir configurer un routeur et les différents composants d'un réseau local. Savoir mettre en oeuvre les aspects fonctionnels et les services applicatifs.

Connaissances préalables nécessaires:

Connaissances de base sur les réseaux et les systèmes d'exploitation.

Programme:

Introduction : Définitions : IP, TCP. Historique. IP dans le modèle ISO.

Protocole IP : Trame, adressage, principes de routage.
Configuration des adresses et des masques réseaux.
Accès à la couche réseau sur différents systèmes d'exploitation.
Configuration de l'interface réseau.

Routage : Interconnexion de réseaux, répéteurs, les ponts. La commutation.
Routeurs et passerelles.
Définition d'une topologie. Principe de routage, algorithmes.
Configuration des routeurs et des postes clients.
Visualisation des chemins utilisés via traceroute.
Routage dynamique : RIP, OSPF.

TCP/UDP : Les protocoles UDP/TCP : mode non connecté/connecté. Connexion virtuelle. Les ports TCP bien-connus (well known ports)

RS003

TCP/IP : protocoles et mise en oeuvre

Applications : Les services du niveau application : telnet, ftp, ssh, scp, traceroute, ping (connexion, transfert de fichiers, contrôle), modèle client-serveur.
Serveurs de noms : DNS (Domain Name System).
Définitions : résolution de noms
Principe : noms de domaines, notion de zones et de responsabilité d'une zone
Architecture : client/serveur
Présentation des notions de serveur primaire, secondaire, cache dns
Arborescence des noms de domaines.
Etude du traitement d'une requête de résolution de nom DNS.
Mise en oeuvre avec bind. Configuration d'un client dns.
Outils d'interrogation : nslookup, host, dig.
Configuration d'un serveur DNS sous Linux.
Etude du fichier named.conf
Analyse des flux et des requêtes client-serveur avec wireshark
Principe d'un serveur DNS secondaire.
SNMP (Simple Network Management Protocol) :
fonctionnalités, apports SNMP V2.

IPv6 : Adressage actuel, attribution des adresses.
Le travail de l'IETF (BradnerMankin). Plan d'adressage sur 128bits.
Agrégateurs : découpage TLA/NLA/SLA/IID. Intégration des Regional Registries
Fonctionnement : Surcharge d'entêtes. Structures des trames. Les nouveaux mécanismes: fragmentation: MTU universelle, DHCPv6, dynamic DNS, renumérotation simplifiée d'un plan d'adressage

Sécurité : Ipsec (IP Security Protocol)
TP de mise en oeuvre

IP: Migration vers IPv6

RS014

Durée: *2 jours*

Prix et dates: nous consulter

Public:
Toute personne souhaitant migrer vers l'adressage IPV6

Objectifs:
Connaître les caractéristiques d'IPV6, et savoir élaborer les méthodes de migration.

Connaissances préalables nécessaires:
Connaissances de bases sur les réseaux et les systèmes d'exploitation.

Programme:

Rappels sur la version 4	: Le protocole IP : trame, adressage, principes de routage. Problèmes d'IPv4.
IPv6	: Structure des trames. Les nouveaux mécanismes : fragmentation : MTU universelle, any cast, renumérotation simplifiée d'un plan d'adressage. Plan d'adressage. Adressage actuel, attribution des adresses. Agrégateurs : découpage TLA/NLA/SLA/IID. Intégration des Regional Registries Entêtes: Mobilité (entête 135), Shim6, sécurité (mise en oeuvre de l'entête calipso), confidentialité et entête d'authentification. Problème des entêtes noeud-par-noeud. Entêtes spécifiques : Fragmentation, Destination
Fonctionnement du multi-cast	: Les groupes prédéfinis. Ajout d'un groupe, inscription. Utilisation du multi-cast dans l'autoconfiguration.
Commandes de base et outils réseau.	: Utilisation des outils de base en IPv6 : wireshark, tcpdump, ping6, traceroute6, ifconfig, nmap, wget, iptraf, netstat, ip6tables, ...

RS014

IP: Migration vers IPv6

Produits	: Supports natifs sur les produits d'infrastructure : messagerie (postfix/dovecot), connexions (ssh), SNMP, NFS, ldap, proxies, ... Supports sur les produits métiers : Web (apache, firefox, IE), tomcat, JBoss, WebSphere
ICMPv6	: Auto-configuration. Découverte des voisins (NDP), découverte des routeurs: fonctionnement, activation, activation partielle, désactivation. Mise en place de radvd. Analyse des trames de découvertes.
Routage	: TP de mise en oeuvre du routage IPv6 en mode statique. Activation du mode automatique, visualisation des tables de routage obtenues.
Gestion des adresses	: Mode sans état, avec état. DHCPv6 : Présentation. Mise en oeuvre d'un serveur dhcpv6. Cohabitation avec IPv4. Attribution statique d'adresses. Gestion du DUID. Stateless Address Autoconfiguration (SAA) : Utilisation de radvd en complément. Relais DHCPv6. Cycle de vie des adresses. Adressage aléatoire. Migration d'opérateurs. Choix de l'adresse client.
DNSv6	: Mise en oeuvre d'un DNS v4/v6. Les différentes implémentations. DNS dynamiques.
Migration v4/v6	: Les différentes approches : double pile, encapsulation statique, encapsulation dynamique. Impacts de la suppression du NAT. Utilisation du cycle de vie des adresses. Les différents tunnels. Mise en oeuvre d'un tunnel 6sur4. Présentation de l'encapsulation v6 dans v4 et l'extension Teredo pour les réseaux à translation d'adresses Comparaison des différentes approches : isanat/Teredo Choix d'un tunnel broker.

RS014

IP: Migration vers IPv6

Cas concret	: Exemple de migration d'une infrastructure complète. La sécurité : IPsec, les pare feux et les filtres. ip6tables, routeurs et répartiteurs de charge, haproxy. Organisation de la migration. Impacts de la mise en oeuvre de plusieurs dhcpv6 et de multiples agents RA. Problème de boucles de tunnels.
Routage dynamique IPv6	: RIP, utilisation en IPv6. Inconvénients du protocole. OSPF v3 pour IPv6 : Présentation du routage des systèmes autonomes. Découverte des routeurs voisins, élection du routeur désigné, calculs des chemins. TP : mise en oeuvre d'OSPF sous Linux avec Quagga.

Messagerie

RS006

Durée: 2 jours

Prix et dates: nous consulter

Public:
Administrateurs réseaux.

Objectifs:
Savoir installer, configurer et administrer une messagerie sous Unix.

Connaissances préalables nécessaires:
Il est demandé aux participants de connaître les notions de base sur les réseaux TCP/IP.

Programme:

TCP/IP	: Mode de fonctionnement: Adressage IP, nommage DNS (serveur de noms) TP : écriture d'un plan d'adressage et mise en oeuvre
DNS	: Fonctionnement, configuration du service.
Concepts de messagerie	: Terminologie : SMTP, POP3, IMAP4, MTA, MDA, MUA... Le routage de messages. Anatomie d'un message, les champs d'entête
Architecture distribuée	: Les clients, modes d'accès au courrier les protocoles : POP, IMAP, principes de fonctionnement. Etude du protocole POP3. Les extensions SMTP. Gestion des pièces jointes. TP : mise en place d'un système complet de messagerie, configuration d'un serveur SMTP (Postfix)
Marché	: Les produits du marché Présentation des serveurs sendmail, Postfix.
Exploitation	: Définitions d'alias. Traitements à l'arrivée (procmail, formail). Anti-spam, clamAV, p3scan. Sécurisation, chiffrage, authentification.

Mise en oeuvre du protocole SNMP

RS022

Durée: *2 jours*

Prix et dates: nous consulter

Public:
Les administrateurs réseau, et toute personne souhaitant mettre en place un système de supervision par SNMP.

Objectifs:
Comprendre le mécanisme de fonctionnement de SNMP, connaitre les outils et produits permettant une utilisation efficace de SNMP dans la supervision du réseau.

Connaissances préalables nécessaires:
Il est demandé aux participants de connaître les bases de TCP/IP.

Programme:

Définitions supervision : Objectifs, méthodes, déterminer les objets à superviser, granularité des tests,
techniques : prélèvements par SNMP, commandes de vérifications, outils spécifiques de supervision.

Le protocole SNMP : Simple Network Management Protocol
Définitions d'objets à superviser, spécifications : RFC 1213.
Historique : depuis SNMP v1, jusqu'aux apports de SNMP v3 (controle d'accès, chiffrement, ..)
Schéma de principe : les requêtes get/set, les agents SNMP.

Fonctionnement : Le principe des MIB. La hiérarchie SNMP.
Les zones privées.
Exemples avec http et ftp.
Détail d'une MIB.
Fonctionnalités :
Exemples : surveillance des différentes ressources d'un poste,
exécution de processus distants

RS022

Mise en oeuvre du protocole SNMP

Mise en pratique	: Commandes d'interrogation des agents SNMP : snmpget, snmpwalk, Notions de communauté et d'Oid (Object Identifier). Configuration d'un agent snmp sous Linux. Exécution de l'agent comme un service. Interrogations simples : description des cartes réseaux du poste client, affichage de la table de routage, ...
Outils d'interrogation	: Graphiques : PTkMib, Mib Browser, MIB Smithy, Automatisation des requêtes avec net-snmp et scli (en mode commande).
Les alertes	: Création d'un serveur d'alertes avec snmptradd. Définition des conditions d'alertes pour chaque objet.
Sécurité	: Authentification Protection du contenu
L'usage de SNMP sur le marché	: Les produits d'analyse, les MIBs développées par les constructeurs.
Développement	: Développement d'une MIB. Présentation des produits de développement. Description de la structure en ASN-1. Travaux pratiques : conversion en C et compilation dans l'agent SNMP, ajout d'OID surveillant la température du processeur, ajout d'OID surveillant le nombre de threads d'un serveur JEE.

Annuaire LDAP

RS122

Durée: 2 jours
1130 €

14 au 15 février
4 au 5 avril
4 au 5 juillet
17 au 18 octobre

Public:

Administrateurs réseaux, intégrateurs d'application souhaitant configurer un annuaire ldap.

Objectifs:

Comprendre le mécanisme de ldap, et savoir mettre en place un service d'annuaire. Les travaux pratiques ont lieu avec Openldap.

Connaissances préalables nécessaires:

Il est demandé aux participants de connaître les bases de TCP/IP.

Programme:

Introduction : le besoin, historique.
Définitions.

Procotole LDAP : Lightweight Directory Access Protocol
Principe de fonctionnement.
Les modèles, la conception d'une arborescence :
construction, importation de schéma

Mise en oeuvre : Travaux pratiques avec OpenLdap :
installation, configuration du serveur.
Les backends openldap.
Définition d'index pour l'optimisation de la recherche dans la base.

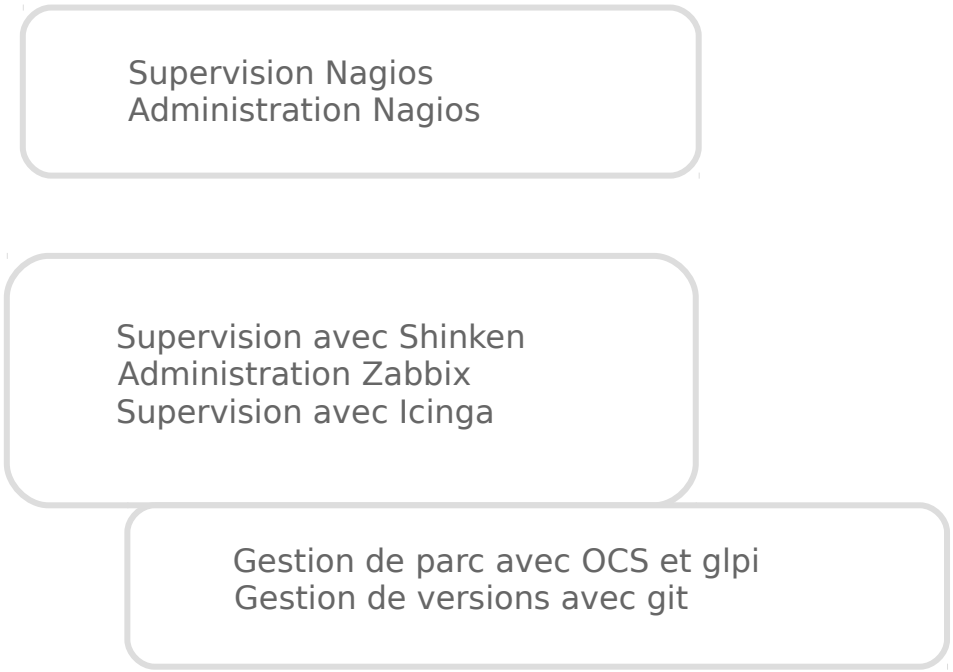
Hiérarchie ldap : Construction de la hiérarchie :
distinguished name, relative distinguished name.
Le format ldif.
Utilisation : commandes de recherche dans l'annuaire (search, compare, add, modify, delete, rename, ...)
Travaux pratiques :
création de fichiers ldif,
ajout à l'annuaire avec la commande ldapadd,
vérification avec la commande ldapsearch.

RS122

Annuaire LDAP

Schéma ldap	: Définitions : attributs, objets Format du schéma. Mécanisme d'héritage des attributs. Organisation d'un schéma. Travaux pratiques : intégration d'un schéma extérieur.
Gestion de l'annuaire	: Outils de création d'un annuaire. Migration de comptes Unix vers Ldap. Importation de fichiers ldif. Méthodes et commandes de consultation dans un annuaire. Travaux pratiques : recherche composée dans un annuaire. Consultation depuis un client de messagerie. Gestion des permissions. Outils graphiques de consultation.
Sécurité	: Authentification, contrôle d'accès, chiffrement des transactions Annuaire et PKI
Architecture	: Distribution, réplication d'annuaires. Le besoin de synchronisation, les méthodes. Travaux pratiques : création d'un serveur esclave, avec réplication de toutes les informations du serveur maître.
Intégration	: Dans le réseau d'entreprise : Exemples : pam/ldap, samba/ldap. Travaux pratiques : création d'un annuaire ldap pour samba, configuration d'u nmodule pam-ldap, Mise en oeuvre pour un serveur de messagerie.
Le marché	: Présentation des principaux annuaires.

Filière Production et supervision



Supervision nagios : utilisation

RS129

Durée: 3 jours
1660 €

18 au 20 février
13 au 15 mai
9 au 13 septembre
2 au 4 décembre

Public:
Exploitants et utilisateurs d'un système de supervision Nagios.

Objectifs:
Connaître les fonctionnalités de Nagios, savoir ajouter de nouveaux tests, savoir mettre en place une politique d'alerte, créer de nouveaux types de notification.

Connaissances préalables nécessaires:
Connaissance de l'architecture d'un système d'information, bases tcpip et bases systèmes.

Programme:

Supervision : définitions	: Les objectifs de la supervision, les techniques disponibles.Objets supervisés.
Les services et ressources	: Rappels sur les principes HTTP, SMTP, NNTP, POP3, PING.Définition des ressources à surveiller.
Présentation de Nagios	: Les fonctionnalités .Supervision, exploitation.Surveillance des services réseaux, Surveillance des ressources (charge CPU, espace disque) Envoi d'alarme vers des contacts déterminés ;Déclenchement de scripts pour corriger les problèmes.
Utilisation	: Les premiers pas avec Nagios : la page d'accueil.Travaux pratiques :utilisation de Nagios pour la supervision d'un ensemble d'hôtes et de services de test. Vue d'ensemble de l'état du réseau.Les hôtes et services.Cartographie du réseau.Visualisation des tests Détection des pannes.Recherche d'hôte.Arrêts programmés

RS129

Supervision nagios : utilisation

Configuration Nagios	: Objets à définir : hôtes, groupes, services, dépendances, notifications, escalades Description des serveurs à surveiller, des contacts, création de groupes de serveurs, de groupes de contacts.Notion de hiérarchie avec les hôtes parents, les dépendances de services, hôtes et groupes.
Déploiement	: Sur les hôtes, principes de NRPE, NSCA.Travaux pratiques :écriture de scripts de déploiement. (NRPE)
Les plugins	: Principe de fonctionnement.Quelques plugins courants
L'interface Centreon	: Les fonctionnalités, les sites de référence, L'architecture Nagios/Centreon.Le positionnement par rapport à Nagios

Administration Nagios

RS130

Durée: 5 jours
2630 €

18 au 22 février
13 au 17 mai
9 au 13 septembre
2 au 6 décembre

Public:

Les administrateurs systèmes, administrateurs réseau, et toute personne souhaitant mettre en place un système de supervision.

Objectifs:

Connaître les fonctionnalités de Nagios. Savoir installer, configurer et administrer le produit. Savoir développer des nouveaux plugins, et mettre en oeuvre la supervision dans un environnement hétérogène.

Connaissances préalables nécessaires:

Notions sur le réseau, bases de TCP/IP. Connaissance d'un langage de script.

Programme:

Supervision : définitions : Les objectifs de la supervision, les techniques disponibles. Rappels sur les principes HTTP, SMTP, NNTP, POP3, PING.Définition des ressources à surveiller.

Présentation Nagios : Les fonctionnalités.Supervision, exploitation.Surveillance des services réseaux, Surveillance des ressources (charge CPU, espace disque).

Architecture : Principe de fonctionnement et positionnement des différents modules.Les plugins et extensions

Installation : Configuration requise. Site de référence.Travaux pratiques : Installation et mise à jour, Paramétrage de base, démarrage Nagios

Utilisation de nagios : Premiers pas avec nagios : la page d'accueil.Vue d'ensemble de l'état du réseau.Détail des hôtes et services.Cartographie du réseau.
Détection des pannes réseau.Les hôtes et services.Travaux pratiques :recherche d'un hôte, arrêt programmé d'hôtes et services.
Liste des vérifications programmées.Edition de rapports.

Administration Nagios

RS130

Configuration	: Etude du fichier de configuration standard nagios.cfg.Description des serveurs à surveiller, création de groupes de serveurs.Description des contacts, et création de groupes de contact, escalades Définition des services et groupes de services.Les notions de hiérarchie, dépendances : hôtes et services.Configuration de l'interface web d'administration. Etude du fichier cgi.cfg
Optimisation de l'ordonnanceur	: Méthode d'ordonnancement.Délai entre chaque test.Entrelacement des services.Tests concurrents.Fréquence de récupération.
Contrôle et débugage	: Analyse des fichiers de logs.Commandes de contrôle.Mode d'exécution des plugins.Options détaillées.
Les plugins	: Principe de fonctionnement.Mise en oeuvre des plugins standards.Travaux pratiques :Personnalisation de Nagios par développement de nouveaux plugins.
Gestionnaire d'évènements	: Mécanisme de traitement d'erreur.Normalisation.Algorithmie de l'ordonnanceur.Macros d'évènements.Démarche d'implémentation. Exemple : relance d'un serveur web.
Lien SNMP	: Présentation du protocole SNMP.Hétérogénéité des superviseurs et du parc supervisé.Tests actifs et passifs.
Supervision distribuée	: Principe des agents.Sur les hôtes, principes de NRPE, NSCA. Travaux pratiques :écriture de scripts de déploiement. (NRPE)installation de nsca et configuration.
Superviseurs redondants	: Méthodes de redondance.La haute disponibilité : mode fail-over, configuration d'un superviseur secondaire, Gestionnaire : panne du superviseur,panne du service nagios.Greffon de test du maître.
Centralisation NDO	: Fonctionnalités et composants.Travaux pratiques :Mise en oeuvre de NDO

RS130

Administration Nagios

Intégration Nagios	: Liens avec Cacti, Centreon, PNP4Nagios. Supervision d'environnements hétérogènes :Windows avec ns_client et check_WMI, Unix (AIX, Solaris), Linux, matériels réseaux (Cisco , Hp), Monitoring d'applicatifs : services web, messagerie, serveurs Jee Nagios et le cloud : supervision intégrée avec OpenStack.

RS135

Supervision avec shinken

Durée: 3 jours
1680 €

4 au 6 février — 23 au 25 septembre
27 au 29 mai — 12 au 14 novembre

Public:
Exploitants et utilisateurs d'un système de supervision Shinken.

Objectifs:
Connaître les fonctionnalités de Shinken, maîtriser l'interface d'exploitation, savoir ajouter de nouveaux tests, savoir mettre en place une politique d'alerte.

Connaissances préalables nécessaires:
Connaissance de l'architecture d'un système d'information, bases tcpip et bases systèmes unix/linux.

Programme:

Supervision : définitions	: Les objectifs de la supervision, les techniques disponibles.Objets supervisés.
Les services et ressources	: Rappels sur les principes HTTP, SMTP, NNTP, POP3, PING.Définition des ressources à surveiller.
Présentation de Shinken	: Historique, licence, fonctionnalités : Supervision, exploitation, surveillance des services réseaux, surveillance des ressources (charge CPU, espace disque),émission d'alertes,actions automatiques programmables, gestion de règles métier. L'architecture : Arbiter, Scheduler, Poller, Reactionner, Broker.Principe d'architecture distribuée. Comparaison avec Nagios.Apports de Shinken.
Installation et configuration	: Prérequis techniques, le référentiel MongoDB. Sur les systèmes Linux, plusieurs méthodes possibles :par le script d'installation, en exécutant setup.py, par les RPMs Travaux pratiques :installation d'un serveur Shinken,démarrage des services.Configuration de l'interface WebUI.Configuration : ajout de widgets, packs,etc Gestion de la configuration :utilisation des templates,Notions de groupes d'hôtes et de services.Autodécouverte avec nmap.

Supervision avec shinken

RS135

Les modules	: Principe, installation de modules.Présentation du site shinken.io Travaux pratiques : mise en oeuvre de modules simples.Installation de l'interface Thruk. Développement de modules en shell et en python.
Performances	: La haute disponibilité avec Shinken.Lissage automatique de la charge par l'architecture de Shinken.

RS150

Zabbix administration

Durée: 3 jours
1680 €

25 au 27 février — 16 au 18 septembre
20 au 22 mai — 18 au 20 novembre

Public:

Les administrateurs systèmes, administrateurs réseau, et toute personne souhaitant mettre en place un système de supervision avec zabbix

Objectifs:

Connaître les fonctionnalités de Zabbix. Savoir installer, configurer et administrer le produit. Savoir mettre en oeuvre la supervision dans un environnement hétérogène.

Connaissances préalables nécessaires:

Notions sur le réseau, bases de TCP/IP. Bases Unix/Linux. Connaissance d'un langage de script.

Programme:

Présentation Zabbix : Historique du produit, version, licence. Systèmes supportés.
Les fonctionnalités de Zabbix :Supervision réseau et serveurs (état des services, charge processeur, disques, ...)
système de configuration d'alertes,interface de supervision,reporting et visualisation des données collectées par zabbix,gestion des ressources (capacity planning)

Architecture : Quelques définitions :notion d'hôte, de groupe d'hôtes, item, trigger, event, action, escalation, media, notification, template
Les composants de l'architecture :Zabbix server, agent, proxy,Java gateway pour la supervision JMX.

Installation : Configuration requise. Plate-formes supportées.Site de référence.
Travaux pratiques :Installation depuis les packages, Initialisation de la base de données.Démarrage du serveur Zabbix.
Configuration depuis l'interface PHP

RS150

Zabbix administration

Configuration	: Gestion des hôtes et groupes d'hôtes,des items, triggers, event.Gestion des notifications sur évènements.Visualisation.Création de templates. Les utilisateurs : configuration, groupes d'utilisateurs,droits d'accès
Applications types	: Supervision de services web,de machines virtuelles,auto-découverte des éléments réseau Travaux pratiques :utilisation de l'interface web
Supervision distribuée	: Principe des proxy Zabbix.Mise en oeuvre d'une architecture distribuée.

RS155

Supervision avec Icinga

Durée: 3 jours
1680 €

18 au 20 février — 30 septembre au 2 octobre
11 au 13 juin — 2 au 4 décembre

Public:
Exploitants et utilisateurs d'un système de supervision Icinga.

Objectifs:
Connaître les fonctionnalités d'Incinga 2, savoir l'installer, le configurer et superviser les services réseaux, gérer les alertes.

Connaissances préalables nécessaires:
Connaissance de l'architecture d'un système d'information, bases tcpip et bases systèmes unix/linux.

Programme:

Supervision : définitions : Les objectifs de la supervision, les techniques disponibles.Objets supervisés.Les services et ressources
Rappels sur les principes HTTP, SMTP, NNTP, POP3, PING.Définition des ressources à surveiller.

Présentation de Icinga : Historique, licence, fonctionnalités : Supervision, exploitation, surveillance des services réseaux,
surveillance des ressources (charge CPU, espace disque),émission d'alertes.Positionnement par rapport à Nagios, shinken

Installation et configuration : Installation et configuration des packages Icinga2,des plugins de supervision.Configuration de la base de données.Installation de l'interface web
Configuration des clients.Différents méthodes:NRPE, ssh, snmp, ...Mise en oeuvre.Configuration de noeuds avec l'agent Icinga 2

Mise en oeuvre de la supervision : Présentation du langage de configuration.Les templates, objets et types de variables.Exemples de commandes de vérification des ressources à surveiller
Intégration de nouveaux hôtes et services.Gestion des alertes et notifications

RS155

Supervision avec Icinga

Compléments graphiques : Présentation des différentes solutions.Démonstration avec PNP4Nagios, Incinga Web 2

Gestion de Parc avec OCS et GLPI

UX124

Durée: 3 jours
1710 €

11 au 13 mars — 16 au 18 septembre
11 au 13 juin — 25 au 27 novembre

Public:
Administrateur devant gérer un parc de systèmes dans un environnement de production.

Objectifs:
Savoir installer et configurer, et utiliser les outils OCS et GLPI.

Connaissances préalables nécessaires:
Une bonne connaissance des systèmes Unix/Linux est nécessaire.

Programme:

Introduction : Le besoin : inventaire et suivi des configurations matérielles et logicielles
Présentation OpenComputer and Software Inventory Next Generation.Fonctionnalités, informations collectées

Architecture OCSNG : architecture client/serveur.Les composants du serveur OCSNG : base de données, serveur de communication, agents, serveur de déploiement, console d'administration
Intégration avec GLPI

Installation : systèmes supportés, installation depuis les packages, création des utilisateurs.Les agents : procédure d'installation et configuration du lancement au démarrage du système

Configuration : configuration par l'interface web : fonction IPDISCOVER, détection des doublons (adresses Mac).notion de TAG : quelques exemples d'utilisation
Les différents paramétres de chaque machine gérée.

Gestion de Parc avec OCS et GLPI

UX124

Utilisation glpi	:	La Console Centrale, l'accès aux différentes fonctionnalités.L'inventaire : requêtes préféfinies.Mise à jour des TAGS, recherche multi-critères, recherche par analyse du TAG,export des données, création de gabarits. Le module Administration : gestion des droits, affectation des logiciels à une catégorie, affectation des ordinateurs à une entité, création de règles, utilisation de dictionnaires.Le module Configuration : définition et modification des composants, configuration de l'affichage, du niveau de journalisation. Notifications par mail.Le module Assistance (help-desk) : suivi des incidents, gestion des appels, des tickets, du planning, des interventions,extraits de statistiques.
Les plugins	:	Principe des plugins.Mise en oeuvre pratique des plugins tracker, reports, racks, data injection, item uninstallation.
Import OCS NG	:	Objectif et principe de fonctionnement.Configuration du mode OCSNG. Options d'importation. Mode d'import OCS.Import OCS par le plugin massocsimport.
Gestion du cycle de vie.	:	Le besoin. Plugin d'injection de fichiers CSV (data_injection).Liaison d'une machine. Plugin de désinstallation d'une machine, de suppression d'une machine. Statuts des matériels. Gestion des machines en stock. Machines en réparation et réformées.
Télédéploiement de paquets	:	Principe et architecture. Notion de priorité et action à exécuter
Exploitation OCS glpi	:	Sauvegardes, journalisation. Lien avec un annuaire ldap.
Fusion Inventory	:	Solution alternative à OCS pour l'inventaire.Principe de fonctionnement avec SNMP. Présentation du plugin FusionInventory et mise en oeuvre.Installation des agents. Lien avec SNMP. Réalisation d'un inventaire réseau.

Gestion de versions avec GIT

UX151

Durée: 2 jours
1175 €

14 au 15 mars — 19 au 20 septembre
17 au 18 juin — 21 au 22 novembre

Public:
Tout développeur, chef de projet, architecte, souhaitant utiliser git comme gestionnaire de versions

Objectifs:
Comprendre les principes d'un gestionnaire de version distribué, les apports de git, savoir le mettre en oeuvre pour gérer les codes sources d'un projet, les versions, corrections de bugs, etc ..

Connaissances préalables nécessaires:
Connaissance des processus de développement et d'un langage de programmation, et des bases Unix/Linux. Les travaux pratiques se déroulent sur Linux.

Programme:

Présentation de Git : La notion de gestionnaire de versions distribué.
Historique de git, licence.
Présentation des principes techniques de git : architecture, les objets stockés
Les différentes utilisations de git :
utilisation d'applicatifs stockés sous git, développement, partage de codes, gestions de modifications, de correctifs etc ...
Aperçu des types de workflows possibles.

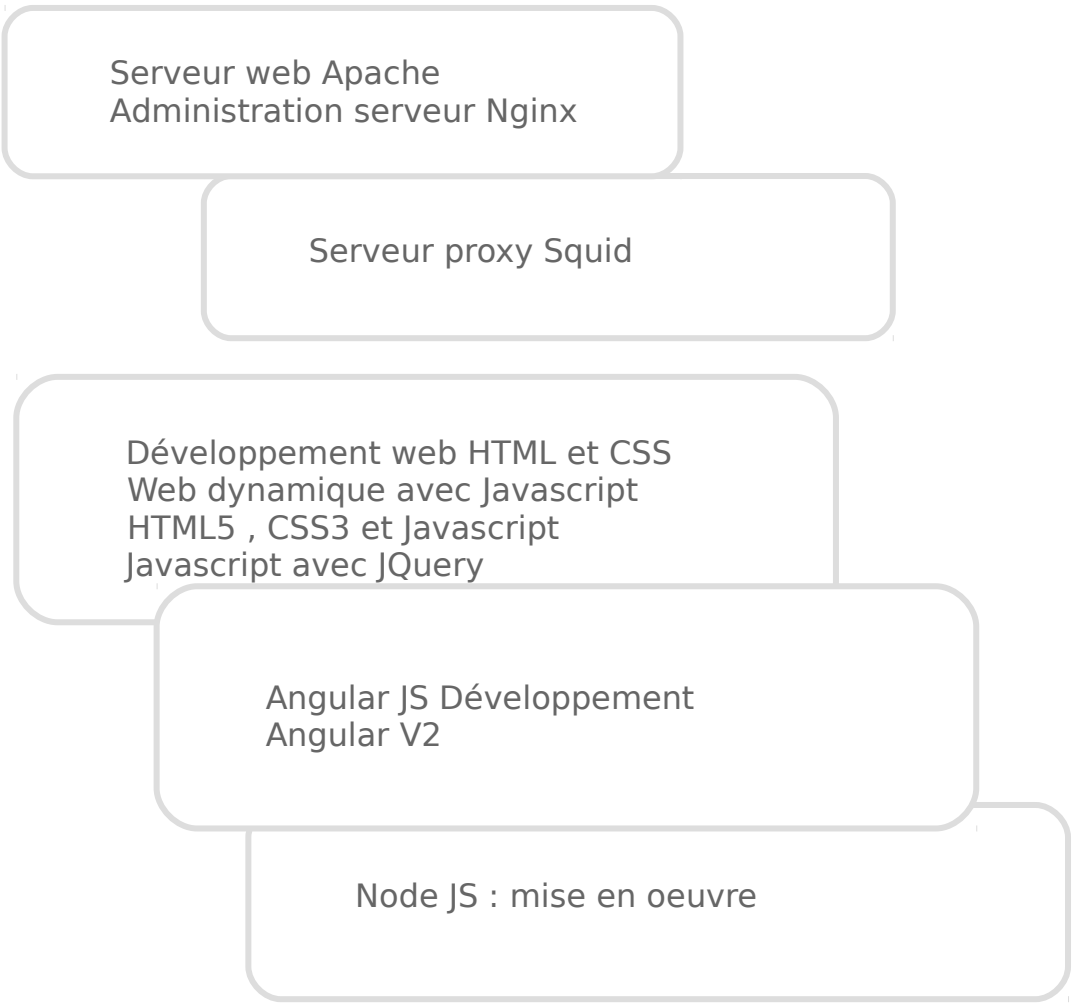
Prise en main : La commande git, options principales
Installation et configuration de git.Présentation des notions de base : référentiel, index, répertoire de travail, clônage
Travaux pratiques :Création d'un premier dépôt.Utilisation de la ligne de commande pour les opérations de base.Enregistrement de modifications simples.Clônage d'un référentiel existant.

UX151

Gestion de versions avec GIT

Gestion des développements	: Etude des commandes principales de manipulation des fichiers :add, status, diff, commit, ... Gestion des branches :branch, checkout, merge, log, stash, etc ... Travaux pratiques :mise en oeuvre sur un projet exemple représentatif des principaux cas d'utilisation Ajout, modification, suppression de fichiers et répertoires.Gestion des commits.Création de branches, navigation entre branches, fusion de branches. Résolution des conflits. Intérêt des branches temporaires.
Travail collaboratif	: Objectif : partage et mise à jour de projets. Fonctionnalités requises : mise à disposition des objets, analyse des modifications, intégration,etc... Définition des rôles (développeurs, intégrateurs).Notion de dépôt local et dépôt centralisé.Etude des commandes : fetch, pull, push, remote, ... Pour le contrôle de fichiers : show, log, diff, ...Gestion des patchs : apply, rebase, revert, ... Travaux pratiques :Connexion à un réferentiel Synchronisation avec un référentiel distant.Utilisation des tags pour identifier des commits.Création et application de patchs sur un exemple de projet complet.
Administration	: Tâches d'administration : nettoyage des arborescences, vérification de la cohérence de la base de données, état du service git Travaux pratiques:Installation d'un dépôt privé centralisé pour une gestion de sources collaborative,import de développements externes avec fast-import
Compléments	: Interagir avec des référentiels partagés via GitHub. Exemples de projets sur GitHub, GitLab Présentation d'outils complémentaires : gerrit, un système de revue de code.Gitweb, l'interface web. GitKraken, client graphique
Bonnes pratiques	: Echanges par rapport aux contextes projets et à l'organistion des équipes pour savoir définir l'utilisation de git la plus adaptée à chaque contexte projet.

Filières Internet et Web



Serveur WEB : apache

RS104

Durée: 3 jours
1660 €

11 au 13 février — 1er au 3 juillet
1er au 3 avril — 14 au 16 octobre

Public:

Toute personne souhaitant configurer, installer et exploiter un serveur web à base d'apache.

Objectifs:

Comprendre le fonctionnement d'apache, et savoir le mettre en oeuvre, l'installer, le configurer et l' administrer.

Connaissances préalables nécessaires:

Il est demandé aux participants de connaître les bases de TCP/IP.

Programme:

Introduction : Présentation, historique. Fonctionnalités.
Documentation de référence.
Fonctionnement multi-threads.
Modularité du noyau.
Travaux pratiques : installation, configuration de base
Principe, le rôle des modules.

Protocole HTTP : Fonctionnement, format des requêtes.
Méthodes. Syntaxe d'une URL.

Configuration du serveur : Environnement, gestion des processus, requêtes, connexions client : le fichier httpd.conf.
Etude du fichier de configuration.
Travaux pratiques :
Configuration des MPM, des DSO.
Connexion des clients. Exemples de dialogue.
Interrogations avec apachectl.

RS104

Serveur WEB : apache

Configuration d'un site principal	: Nom interne du serveur, rappel sur les DNS, page d'accueil, types de fichiers, les alias, définition de chemins relatifs, la balise index Ports et interfaces d'écoute. Distribution des documents. Documents par défaut et chemins relatifs. Travaux pratiques : mise en oeuvre avec création d'un site et configuration du serveur apache.
Hôtes virtuels	: Principe, configuration, hôtes virtuels basés sur l'adresse IP, sur le nom
WebDynamique	: Principe des scripts CGI, fonctionnement.Apports de la solution fastCGI. Travaux pratiques : configuration apache, écriture d'un script CGI simple. Présentation des variables d'environnement disponibles. Passage de paramètres avec GET et POST
Exploitation Apache	: Administration du site, personnalisation des pages d'erreurs Les fichiers journaux: analyse directe, analyse par webalizer Travaux pratiques : mise en charge du site et visualisation du comportement.
Le mode proxy	: Architecture forward proxy et Reverse proxy. Travaux pratiques : installation et configuration. Gestion du cache avec le module mod_cache.

Administration serveur Nginx

RS105

Durée: 3 jours
1660 €

4 au 6 février
13 au 15 mai
2 au 4 septembre
9 au 11 décembre

Public:
Toute personne souhaitant configurer, installer et exploiter un serveur Nginx.

Objectifs:
Comprendre le fonctionnement du serveur Nginx, et savoir le mettre en oeuvre, l'installer, le configurer et l'administrer.

Connaissances préalables nécessaires:
Il est demandé aux participants de connaître les bases de TCP/IP et des technologies web.

Programme:

Introduction	: Présentation, historique, licence. Fonctionnalités : serveur http, proxy inverse, proxy de messagerie, diffusion de contenus vidéo, ... Plates-formes supportées. Architecture : principe de serveur asynchrone, modularité.
Mise en oeuvre	: Choix des modules, d'une version/distribution. Travaux pratiques : installation, démarrage, configuration de base
Protocole HTTP	: Fonctionnement, format des requêtes. Méthodes. Syntaxe d'une URL.
Configuration du serveur nginx	: Etude du fichier /etc/nginx/nginx.conf : http-block, server-block, location-block.
Configuration d'un site principal	: Nom interne du serveur, rappel sur les DNS, page d'accueil, types de fichiers, les alias, définition de chemins relatifs, Travaux pratiques : mise en oeuvre avec création d'un site et configuration du serveur Nginx.

RS105

Administration serveur Nginx

Virtuals Hosts, locations	: Principe, configuration, hôtes virtuels basés sur l'adresse IP, sur le nom. Configuration des URLs.
Scripts CGI	: Exécution d'applications en PHP, en Python.
Serveur proxy	: Configuration Nginx en proxy inverse et en proxy messagerie.
Exploitation	: Administration du site. Les fichiers journaux : création de fichiers de logs séparés pour chaque hôtel virtuel. Mécanisme de rotation des fichiers journaux. Analyse des informations stockées dans les logs. Travaux pratiques : mise en charge du site et visualisation du comportement.
Sécurité	: Le module HttpAuthBasicModule. Mise en oeuvre des directives auth_basic et auth_basic_user_file. Création des login/mot de passe : htpasswd-b ou htpasswd-perl. Restriction d'accès en fonction de l'adresse IP : directives allow et deny, et avec restriction par mot de passe : directive satisfy.
Migration	: Comparaison des serveurs Nginx et Apache. Conseils de migration.

Serveur Proxy Squid

RS114

Durée: 2 jours

Prix et dates: nous consulter

Public:
Toute personne souhaitant mettre en place un serveur Squid afin d'optimiser et sécuriser les accès Internet de l'entreprise.

Objectifs:
Comprendre les principes de base d'un serveur de cache Internet. Savoir élaborer les configurations optimisant le fonctionnement du serveur Squid. Mettre en place les contrôles d'accès dans une architecture solide et sécurisée.

Connaissances préalables nécessaires:
Des connaissances minimales sur TCP/IP (adressage, fonctionnement) ainsi que sur le fonctionnement du Web sont nécessaires pour suivre ce cours.

Programme:

Besoin	: Serveur Proxy, fonctionnement, multi serveurs proxys, hiérarchie de serveurs cache, cache transparent, accélérateur Web ou proxy inversé.
Squid	: Présentation, sécurité, architecture externe.Exemple d'utilisation, systèmes d'exploitation concernés, logiciels complémentaires.
Installation	: Installation à partir de paquetages, fichiers de configurations, configuration de base, test du serveur.
Configuration des postes clients	: configuration manuelle, automatique. Scripts d'auto-configuration, filtrage suivant DNS, par protocole. Clients en mode texte,robots. Installation dans le navigateur.
Configuration du serveur	: principe et syntaxe des ACL. Optimisation de l'utilisation du serveur. Restriction d'accès par hôte, par réseau, par plage horaire, par jour, par site. Mise en cache des données. Méthodes d'authentification.
Administration	: Surveillance, support SNMP. Configuration par WebMin. Fichiers journaux

Serveur Proxy Squid

SquidGuard : Présentation, les groupes source, groupes de destination. Reécriture d'URL, règles d'accès. Principe de la base de données, utilisation, considérations de performances

Développement Web HTML et CSS

RS102

Durée: 2 jours
1120 €

4 au 5 février
13 au 14 mai
23 au 24 septembre
12 au 13 novembre

Public:

Utilisateurs de l'informatique, contributeurs, informaticiens (chefs de projet, concepteurs, réalisateurs), webmestres amenés à participer à l'élaboration de documents en vue d'une publication sur le Web

Objectifs:

Maitriser les fonctionnalités de base d'HTML5 pour concevoir des pages webmestres

Connaissances préalables nécessaires:

Connaissances générales sur Internet.

Programme:

Introduction : Contexte historique du web, présentation du langage HTML, bon usage des balises
Les nouveautés du HTML5

Contenu et présentation : Texte simple,titres,paragraphes
Mise en forme, caractères spéciaux
Listes, séparateurs, autres balises de texte
Les iframe

Les tableaux : Utilisation, structure
Les cellules, fusion des cellules
Titre et légende, en-tête
Groupes de colonnes/lignes

Les liens : Balise HTML
Liens externes, liens internes
Liens de mail, liens vers des fichiers
Attributs, couleurs

Les images : Les couleurs en HTML
Formats d'images du Web
Insertion d'une image, attributs, arrière plan

RS102

Développement Web HTML et CSS

Les formulaires	: Structure HTML Types de champs: texte de saisie, listes, cases à cocher Champs cachés, mot de passe Envoi de formulaire GET, POST Transfert de fichier Validation
Feuilles de styles CSS	: Objectifs,syntaxe Style interne/externe Types de sélecteurs (Balise, ID, Classe) Les pseudo-classes Modèle de boites, positionnement, décoration Séparation contenu présentation
En-tête HTML	: Le DOCTYPE, le rôle des balises, balises meta importantes Conseils pour le référencement
Multimédia	: Présentation, fichiers audio, fichiers vidéo Animation Flash
JavaScript	: Le code côté client/serveur Présentation,scripts internes et externes Contenu du document, evénements Fonctions utiles, traiter un formulaire

Web Dynamique avec JavaScript

RS106

Durée: 3 jours
1700 €

21 au 23 janvier
11 au 13 mars
1er au 3 septembre
14 au 17 novembre

Public:
Développeurs web, architectes web, chefs de projets, webmestres

Objectifs:
Comprendre et maîtriser le langage JavaScript

Connaissances préalables nécessaires:
Il est demandé aux participants de connaître le principe de fonctionnement d'Internet et du Web, le langage HTML. La connaissance d'un langage de programmation est utile.

Programme:

Présentation	: Historique et évolution du langage. Evolution de l'utilisation du langage Organisation du code.Outils de développement. Principes de base HTML et CSS.Les règles, sélecteurs, propriétés de styles, etc .. Interaction avec Javascript.
Syntaxe Javascript	: Les variables, les types (Number, String, Boolean, ...). Tableaux, boucles et tests. Les opérateurs arithmétiques et logiques. Travaux pratiques :réalisation d'exemples simples Gestion des erreurs et des exceptions,exemples de mise en oeuvre des instructions 'try', 'catch', 'throw', 'finally' Utilisation de la console.Méthodes et outils de debugging. Présentation des fonctions globales et des classes natives. Définition des fonctions.Gestion des arguments
Utilisation du DOM	: Présentation du Document Object Model (DOM). Fonctions de sélection, fonctions de création d'objet DOM Modifier les éléments du DOM. Travaux pratiques :exemple de validation d'un formulaire

Web Dynamique avec JavaScript

RS106

Gestion des évènements	: Principe et définitions.Présentation des évènements courants. Flux évenementiel du DOM.Lier un évènement à un objet du DOM. Intéragir avec les éléments du DOM. Travaux pratiques :exemple d'un gestionnaire d'événement générique L'objet 'event'.Les types d'événements à gérer.Bonnes pratiques.
AJAX : Asynchronous JavaScript And XML	: Présentation et exemple d'utilisation
Déroulement d'une requête AJAX	: Protocole utilisé, limites. Détails de l'objet XMLHttpRequest Travaux pratiques :Initialisation d'une requète AJAX et utilisation Contourner les limitations
Programmation Orientée Objet	: Définitions de la POO. Utilisation de la POO en Javascript. Plusieurs façons de créer un objet en Javascript. Améliorer la création d'un objet avec « prototype » Emuler un singleton en javascript Travaux pratiques :exemple de création d'objets 'inline'
Fonctions avancées en Javascript	: Utilisation du mot clé « this » Les closures, définition, cas d'utilisation. Méthodes apply et call
Présentation des nouvelles fonctionnalités liées à HTML 5	: Nouvelles balises (vidéo, son, ...) L'API File, les websockets, les workers, le webstorage

HTML5 CSS3 et javascript

RS107

Durée: 3 jours
1650 €

4 au 6 février
13 au 15 mai
23 au 25 septembre
12 au 14 novembre

Public:
Développeurs web, architectes web, chefs de projets, webmestres

Objectifs:
Maîtriser les fonctionnalités avancées d'HTML5 JavaScript et CSS3 pour concevoir des applications webmestres

Connaissances préalables nécessaires:
Bonne connaissance du langage javascript.Connaissance des fondamentaux internet et HTML

Programme:

HTML5	: Nouveautés, doctype Eléments syntaxiques et nouvelles balises Les nouveaux attributs, evolution des formulaires Découverte d'Ajax Level 2, les microformats, les attributs data
API Javascript	: Sélecteurs CSS, usage des Timers Les Workers et l'API Message
Web Storage	: Session, local. Database et IndexedDB
Drag and Drop	: Content Editable et commandes. Ofline web application. Géolocalisation, web Socket, device API
CSS3	: Présentation, les sélecteurs, pseudo classes Media Queries et responsive design
Mise en forme	: Polices exotiques, ombrages, transparence, dégradés Propriétés display
Transitions	: Transformations, animations
API mobile	: JQuery Mobile

RS108

Javascript avec JQuery

Durée: 3 jours
1700 €

28 au 30 janvier — 2 au 4 septembre
20 au 22 mai — 9 au 11 décembre

Public:
Développeurs web, architectes web, chefs de projets, webmestres

Objectifs:
Développer en JavaScript avec jQuery

Connaissances préalables nécessaires:
Très bonne pratique du langage Javascript. Connaissances des fondamentaux internet et HTML

Programme:

Présentation de JQuery	: Installer JQuery.Conseils sur les performances.Première utilisation La fonction $() ou jQuery(), les sélecteurs jQuery.Sélecteur CSS, sélecteur d'attribut, sélecteur personnalisés.La méthode .filter(), autres méthodes de parcours du DOM
Manipuler les éléments du DOM	: Accéder directement à un élément du DOM.Modifier les balises dynamiquement
Les évènements en jQuery	: Description des évènements.Évènements au chargement de la page, evènements associés au DOM.Attacher un évènement sur un élément du DOM.Méthode .bind(), .live(), .delegate(), .on() Les raccourçis d'évènements, les callBacks.callBack avec et sans arguments.Gérer les évènements multiples, propagation des évènements.Retirer un évènement sur un élément du DOM.Les évènements et les espaces de nom
AJAX avec JQuery	: Premiers pas, les paramètres de la méthode .ajax().callBack.done(), callBack.fail() Les webStorage et éléments éditables
Les effets visuels	: Effets visuel intégrés à jQuery.Animations personnalisées

Angular JS développement

RS110

Durée: 3 jours
1700 €

18 au 20 mars
27 au 29 mai
14 au 16 octobre

Public:
Développeurs web, architectes web, chefs de projets, webmestres

Objectifs:
Comprendre les principes de AngularJS, savoir développer une application web et utiliser les bonnes pratiques de développement.

Connaissances préalables nécessaires:
Très bonne pratique du langage Javascript. Connaissance des fondamentaux internet et HTML

Programme:

Introduction	: Historique et principes généraux. Pattern MV*, data-binding bi-directionnel.Avantages et inconvénients Travaux pratiques: installation et découverte.Développement application « Hello world! »
Modèles, vues et contrôleurs	: Le modèle. Les vues : templates,expression,directives,filtres Les contrôleurs : fonctionnement,le $scope
Les modules	: Création et configuration, partage de services.Injection de dépendances
Le routage	: « Single page application ». Configuration, méthode .config du module.Paramètres d'url, evénements de routage
Les formulaires	: Lien entre ng-model et contrôleur. Validation, états des formulaires, gestion d'erreur.Types de champs, classes CSS, custom validation
Les directives et les filtres	: Convention de nommage, écriture normalisée. Types de composants, directives de templates Options de configuration, transclude, scope, cycle de vie. Fonctions compile et link, créer ses propres filtres

RS110

Angular JS développement

Cycles de vie et scopes	: Présentation du fonctionnement d'AngularJS. L'arbre des scopes,les méthodes de $rootscope.Les événements : diffusion et interception
Les services AngularJS	: Les services natifs, les values et contants. Factory, Service et Provider
AJAX et REST	: Le service « $http », le service « $resource ».L'API de promise
Aller plus loin	: Bonnes pratiques.Internationalisation, angular-translate.L'interface, l'initialisation d'un projet
Tester une application AngularJS	: Tests unitaires : Karma, Jasmine. Tests end-to-end (e2e) : Protractor.Outils de debug ng-inspector

Développer une application web avec Angular

RS112

Durée: 3 jours
1700 €

23 au 25 avril
24 au 26 juin
16 au 18 septembre
18 au 20 novembre

Public:
Développeurs web, architectes web, chefs de projets, webmestres

Objectifs:
Savoir développer des applications avec le framework Angular, comprendre l'architecture d'angular, découvrir l'architecture typescript

Connaissances préalables nécessaires:
Bonnes connaissances de Javascript

Programme:

Présentation	: Positionnement d'Angular. Angular vs AngularJS.Rôles de Typescript et ES6.Aperçu de l'architecture.Présentation d'Angular CLI Découverte d'un exemple minimal.La phase de démarrage
Outillage	: Débogage dans le navigateur.Augury.Visual Studio Code
TypeScript et ES6	: Transpilation.Let et const, Template strings, Typage, Classes et interfaces.Les modules.Arrow functions.Décorateurs
Les composants	: La notion centrale d'Angular.Structure d'une application Angular.NgModule.Structure d'un composant.Template.Styles.Création d'un composant avec Angular CLI Cycle de vie des composants.Imbrication des composants.Content projection
Templates	: Template et DOM.Interpolation et expressions.Binding et interactions @Input et @Output.Binding bidirectionnel.La notion de directives.Directive de structure et directive d'attribut NgClass, NgStyle et NgModel.NgFor, NgIf et NgSwitch
Injection de dépendances	: Principes.Les services et @Injectable. Injectors et providers. Les types de providers.Token et @Inject

Développer une application web avec Angular

RS112

RxJS et Observables	: Principes de la programmation réactive.La librairie RxJS.La notion de flux.Les 'Observables'.Quelles utilisations dans Angular ?
Routage	: Importance du routage.Configuration du RouterModule.RouterOutlets.Définition des Routes.Naviguer vers une route.Routes secondaires Routes paramétrées
Appels HTTP	: La notion de services HTTP.Les APIs proposées : Http et HttpClient.Envoi de requêtes
Formulaires	: Structure d'un formulaire.Formulaires par template.Validations.Gestion des styles.Formulaires réactifs.FormControl et FormGroup Groupe de champs avec FormBuilder.Gestion des modifications

Node JS mise en oeuvre

RS111

Durée: 3 jours
1700 €

25 au 27 février
23 au 25 avril
1er au 3 juillet
7 au 9 octobre

Public:
Développeurs web, architectes web, chefs de projets, webmestres

Objectifs:
Développer en javascript côté serveur avec Node.js. Comprendre les principes de Node.js et utiliser les bonnes pratiques de développement.

Connaissances préalables nécessaires:
Connaître les fondamentaux internet.Savoir programmer en javascript.

Programme:

Introduction	: Historique, principes généraux.Fonctionnement interne.Exemples d'applications.Avantages et inconvénients Installation et découverte.Application « hello world »
Les modules	: Se servir des modules de base, NPM
Serveur web	: Ma première application web. Gérer les requêtes et les réponses HTTP. Routage des URLs. Opérations bloquantes et non-bloquantes
Express	: Paramétrage, gestion des requêtes HTTP. Sessions, templating
Autres composants	: Socket.IO, connect, async
Tests	: assert / expect.js, Mocha / Zombie, PhantomJS / CasperJS
Performances	: single-thread et event-loop. Communication inter-processus.Redis, le module « cluster »

Filières Développement

Programmation en langage C
Perfectionnement en langage C
Programmation système en C
Programmation noyau et drivers en C

Développement web avec PHP
Programmation Perl
Le langage Go

Le langage Python
Développement python avancé
Python web avec Django

Introduction à XML
Développement applications android

Programmation en langage C

LC001

Durée: 5 jours

Prix et dates: nous consulter

Public:
Tout développeur souhaitant apprendre le langage C.

Objectifs:
Connaître et maîtriser les concepts de base du langage C. Savoir écrire des programmes simples et acquérir des méthodes de programmation.

Connaissances préalables nécessaires:
Connaissance d'un langage de programmation.

Programme:

Le C	: Présentation du langage C. Avantages et inconvénients. Architecture, syntaxe. Structure d'un programme C. Compilation.
Contrôle de programme	: Instructions de contrôle. Boucles for et while. Les tests. Les branchements avec break, continue, return, exit, goto et switch.
Manipulation de données	: Types de données. Les variables, tableaux, chaînes de caractères. Déclarations de variables. Utilisation des types: variables entières, réelles, structurées. Les types primitifs : char, short, int, long, float, double. Type statique, registres. Manipulation des tableaux : initialisation, accès aux tableaux. Traitement des chaînes de caractères : initialisation, saisie, accès Notions sur les variables externes. Conversion, règles de portée.

LC001

Programmation en langage C

Fonctions	: Fonctions de base : affichage et lecture des données. Entrées/sorties formatées : options d'affichage des caractères, entiers, ... Fonctions spécifiques aux chaînes de caractères : strcpy, strcat, strchr, strupr, strlwr, strlen. Fonctions personnalisées : définitions, règles de fonctionnement, récursivité.
Programmation structurée	: Notion de structure. Les unions, champs binaires, types énumérés. Définition, déclaration, utilisation de srtuctures. Exemples d'utilisation des champs binaires et mise en oeuvre. Déclaration d'énumération avec enum. Etude d'exemples.
Les opérateurs.	: opérateurs de calcul, simplification d'écriture, opérateurs de décalage (>> et <<), et binaires(| ^). Erreurs de conversion implicite. Opérateurs de comparaison (== <= >= ? ...) et opérateurs logiques (! ||) Priorités des opérateurs.
Librairies	: Introduction à la notion de librairie. La librairie standard. les fichiers inclus. Introduction aux différentes phases de compilation, édition de liens.
Allocation dynamique	: Présentation, les pointeurs. Principe de l'allocation dynamique. Applications. Exemples des listes chaînées et arbres binaires. Pointeurs sur les fonctions.

Perfectionnement en langage C

LC002

Durée: 5 jours

Prix et dates: nous consulter

Public:
Toute personne amenée à programmer, à superviser ou à modifier des logiciels écrits en langage C.

Objectifs:
Compléter des connaissances en langage C par une formation approfondie sur les mécanismes fondamentaux de fonctionnement.

Connaissances préalables nécessaires:
Il est demandé aux participants de connaître les structures et fonctions de base du langage C.

Programme:

Rappels : Les variables, type statique.
Variables statiques et variables registres.
Conversions.

Structures : Présentation, intérêt des structures.
syntaxe de définition, déclaration et d'accès aux éléments des structures.
Exemples: copie de structures
Structures avancées, unions, tableaux, champs binaires, drapeaux.

Fonctions de saisie, affichage : Options avancées de printf.
Mise en forme paramétrée.
Saisie avancée.
Rappel du principe : décomposition du flux d'entrée.
Les types 'ensemble'.

Les opérateurs : Rappels sur les opérateurs de calculs.
Opérateurs logiques, opérateurs binaires.
Travaux pratiques :
mise en oeuvre des opérateurs de décalage.
Priorité des opérateurs.

LC002

Perfectionnement en langage C

Fonctions	: Pointeurs sur les fonctions. Applications aux interpréteurs.
Allocation dynamique	: Principe d'allocation mémoire. Syntaxe de malloc et free. Travaux pratiques de mise en oeuvre. Fonctions avancées (calloc et realloc) : intérêt et applications.
Pratique	: Les listes chaînées, les arbres binaires. Applications à l'organisation des données.
Techniques de programmation	: Les phases de compilation : précompilation , assemblage, édition de liens. Définition de constantes. Contrôle de compilation. Les macro-instructions. Conventions de nommage. Comparaison avec les fonctions. Les fichiers inclus : #include.
Bibliothèques	: Méthode, syntaxe Les bibliothèques standards : libc.a, libm.a, libcur.a Fonctions disponibles dans la bibliothèque mathématique.
Les entrées/sorties	: Mécanisme de stockage des fichiers. Méthode d'accès, les descripteurs de fichiers. Fonctions open/close. Travaux pratiques : écriture d'une fonctions permettant de tester l'existence d'un fichier. Fonctions read/write. Mise en oeuvre avec lecture/écriture de structures. Modes d'ouvertures spécifiques : avec positionnement dans le fichier, avec création du fichier, ... Options : O_TRUNC, o_SYNC, O_NDELAY. Le type FILE : mise en oeuvre de fprintf, fscanf, fgets, fputs.

Programmation système en C sur Unix/Linux

LC010

Durée: 3 jours

Prix et dates: nous consulter

Public:

Toute personne amenée à programmer, à superviser ou à modifier des logiciels écrits en langage C et liés au système d'exploitation.

Objectifs:

Compléter des connaissances en langage C par une formation approfondie sur les mécanismes d'accès au système d'exploitation. L'accent sera particulièrement sur les fichiers, pointeurs, allocations de mémoire, communications et les librairies systèmes.

Connaissances préalables nécessaires:

Il est demandé aux participants de bien connaître les structures et fonctions de base du langage C.

Programme:

Rappels	: Architecture d'un programme écrit en C.Phases de compilation.
Gestion de la mémoire	: Rappel sur l'organisation de la mémoire.L'adressage par les pointeurs.Les opérateurs et *. Les pointeurs et les arguments de fonctions.Les calculs d'adresses.Les fonctions d'allocation malloc et free,et les appels systèmes: sbrk, realloc. Travaux pratiques :écriture d'un allocateur de mémoire.

Programmation système en C sur Unix/Linux

LC010

Communications inter-processus. : Les différentes méthodes : pipes, fifo, signaux, files de messages.Signaux et interruptions : les principaux signaux.
Travaux pratiques :émission d'un signal avec kill(), réception du signal par signal().Sémaphores et appels concurrents :principe de fonctionnement des sémaphores.Travaux pratiques :mise en oeuvre avec semget, semctl, semop.Segments de mémoires partagées :définitions de constantes et structures,
Travaux pratiques :création d'un segment de mémoire partagée avec shmget,
attachement, détachement d'un segment avec shmat, shmdt.Files de messages :constantes et structures nécessaires pour la manipulation des files de messages.
Travaux pratiques :mise en oeuvre de la primitive msgget(),gestion des files de messages (consultation, modification, suppression) avec msgctl()
Envoi d'un message à une file : msgsend().Segments partagés :définition d'un segment de mémoire partagé. Description et mise en oeuvre des appels systèmes shmat(), shmget().
Utilisation de sémaphores pour la gestion des accès concurrents au segment.Sockets BSD :mise en oeuvre des prises réseaux pour la communication interprocessus.
Exemple avec des liens locaux. Extension aux liens distants. Communications inter-machines.

Les processus et la parallélisation : Création de processus.Définition et mise en oeuvre des primitives fork(), clone(), setsid().
Limites d'utilisation. Introduction aux threads.Les threads. La norme et les implémentations.
L'implémentation Posix : NPTL. Cycle de vie des threads: création, destruction. Synchronisation entre threads, détachement du processus principal, attente de fin d'exécution.
Attributs des threads. Gestion de la mémoire consommée, gestion de la pile de données.Gestion des accès concurrents, principe de l'exclusion mutuelle.
Travaux pratiques :mise en oeuvre des mutex.Coopération de traitements entre threads. Mise en oeuvre des conditions variables. Gestion des signaux dans un thread.Ordonnancement de threads.

Programmation noyau et drivers en C sur Linux

LC011

Durée: 3 jours

Prix et dates: nous consulter

Public:
Tout développeur souhaitant gérer les modules du noyau, ou en programmer de nouveaux.

Objectifs:
Comprendre le fonctionnement des modules dans le noyau, et savoir concevoir des drivers.

Connaissances préalables nécessaires:
Connaissance du système d'exploitation Linux, maîtrise de la programmation en langage C.

Programme:

Architecture : Architecture d'un système Unix. Mode user, mode kernel. Logs.
Notion de pilotes/drivers.Architecture d'un système Linux.
Notion de modules. Les distributions : desktop, embarquées (openWRT).
Gestion des modules: ajout, suppression, paramétrage.
Travaux pratiques :ajout d'un module simple sur une distribution standard.
Présentation de la buzybox pour les distributions embarquées.

Compilation noyau : Compilation d'un noyau pur officiel. Description de la chaîne de compilation.
Options de compilations. Mise en place du nouveau noyau.
Travaux pratiques :modification d'un module pilote. Ajout au noyau précédent.Création d'un module de base.

Programmation de modules : Principes fondamentaux : timers et alarmes, journalisation, échanges de données kernel-mode/user-mode, interactions dynamiques avec un module,
passage de paramétres, gestion des tâches task_struct, allocation mémoire kmalloc, verrouillage du noyau en cas d'accès concurrents (lock_kernel).

Le langage Go

LG001

Durée: 4 jours
2200

18 au 21 mars
1er au 4 juillet 7 au 10 octobre

Public:
Tout développeur souhaitant apprendre la programmation avec le langage Go.

Objectifs:
Connaître et maîtriser les concepts de base, savoir écrire des programmes simples en Go et mettre en oeuvre les mécanismes de programmation multi-thread

Connaissances préalables nécessaires:
Connaissance d'un langage de programmation structuré.

Programme:

Présentation	: Historique de Go,objectifs des fondateurs, positionnement par rapport aux autres langages Particularités techniques : programmation multi-threadind, simplicité Aspects compilation et gestion de la mémoire. Documentation de référence pour les développeurs.
Premiers pas en Go	: Prérequis système Outils, installation de l'environnement de développement Création d'un programme simple en Go : "Hello world"
Fondamentaux	: notions de packages et d'imports,les variables, types de base, conversion de types, constantes, ... Instructions de contrôle (boucles, tests, etc ... :for, if,else, switch , defer Mise en pratique : exemples boucles et fonctions,
Autres types	: Pointeurs, structures, tableaux.Notion de slices et maps.Exemples et exercices de mise en oeuvre

LG001

Le langage Go

Methodes et interfaces	: Définition des méthodes en Go,les pointeurs et fonctions, Définition des interfaces, implémentations.Les types assertions et types switch Exercices de mise en pratique
Programmation concurrente	: Présentation des goroutines : principe et exemple de fonctionnement. Notion de channel. Exemple de programmation sur une architecture multi-processeurs
Compléments	: Quelques packages utiles : json, gobs, reflect, image, image/draw Outils : debuggoing de code avec GDB Data Race Detector, Godoc pour la documentation,outils d'optimisation de code

Développement d'applications Web avec PHP

LH001

Durée: 5 jours

Prix et dates: nous consulter

Public:
Développeurs, concepteurs/réalisateurs, webmestres, chefs de projet Web, architectes techniques.

Objectifs:
Maîtriser la syntaxe du langage. Développer des applications Web dynamiques en PHP. Comprendre l'orientation web de PHP. Gérer des formulaires et les accès aux données. Gérer les utilisateurs de l'application.

Connaissances préalables nécessaires:
Connaissances de base web et html, bases de données Sql. Pratique de la programmation objet. Connaissances des concepts des architectures multi-tiers.

Programme:

Introduction	: Vue d'ensemble de PHP. Structure de base d'une page PHP. Règles de nommage. Installation de PHP (mécanismes d'installation).
Syntaxe de base du langage	: Variables, constantes, types, tableaux.Fonctions. Opérateurs.Gestion des chaînes de caractères, des dates, des nombres.
Les structures de contrôle	: Les contrôles conditionnels : if et switch.Les contrôles itératifs : while, do while, for, continue.Inclusion de fichiers : fonctionnement et utilisation. Comment sortir d'une structure de contrôle.Interruption d'un script.
Introduction aux concepts objet	: Concept Objet. Fonctions et classes.Gestion des exceptions.
Gestion des formulaires	: Récupérer les informations du formulaire.Construction de l'interface utilisateur.Contrôles.Gestion des codes et pages d'erreur.Les filtres.

LH001

Développement d'applications Web avec PHP

Connectivité avec les SGBDR	: Les principaux SGBDR.SQLLite et MySQL.Connexion et déconnexion.Lire et mettre à jour les données. Gérer les erreurs. Requêtes préparées.
La gestion des sessions.	: Fonctionnement des sessions.Débuter une session et assigner des variables.Suppression des variables de session. Destruction d'une session. Les cookies.
PHP et le système de fichiers.	: La sécurité sur les fichiers.Les fonctions de manipulation d'images.Création de graphiques (JPGRAPH, ARTICHOW).Génération de fichiers Excel.Les concepts de sécurité, l'authentification.

Programmation Perl

LP001

Durée: 3 jours
1565 €

11 au 13 mars
22 au 24 mai
16 au 18 septembre
2 au 4 décembre

Public:
Tout développeur souhaitant acquérir les bases de la programmation en Perl.

Objectifs:
Comprendre les principes de base de Perl, connaître la syntaxe de base.

Connaissances préalables nécessaires:
La connaissance d'un langage de programmation sera appréciée.

Programme:

Introduction : Présentation de Perl : caractéristiques, positionnement par rapport à d'autres langages (C, Java, Shell, PHP).Plates-formes d'utilisation.Installation de Perl. Exécution d'un programme Perl. Quelques règles utiles.

Les bases : syntaxe, structure des programmes.Les données. Déclaration de variables.Opérateurs de liaison, de décalage, ...Manipulation de scalaires

Les fonctions : : Principe, appel d'une fonction, passage des arguments,renvoi d'une valeur.Appel d'une fonction. Visibilité des variables.Les références : définition, création de références, références symboliques
Prototype. Fonctions prédéfinies

Structure d'un script Perl : Les tests, boucles. Opérateurs de contrôle, modificateurs

Expressions régulières et variables spéciales : Les expressions régulières, les méta-caractères.Motifs particuliers. Quantificateurs et classes.Opérateur de substitution de motif, de lettre. Variables spéciales. Constantes particulières

Tableaux : Manipulation de tableaux, hachage, sauvegarde des tableaux

LP001

Programmation Perl

Fichiers : Entrées-sorties standards. Manipulation de fichiers.Les redirections. Opérateurs de test de fichiers

Exécution : différentes méthodes d'exécution :compilation , exécution.

LY001

Le langage Python

Durée: 4 jours
2235 €

11 au 14 mars
3 au 6 juin
23 au 26 septembre
25 au 28 novembre

Public:
Tout développeur souhaitant acquérir les bases de la programmation en Python.

Objectifs:
Connaître les possibilités du langage Python, maîtriser les techniques de programmation et apprendre les bonnes pratiques de développement.

Connaissances préalables nécessaires:
La connaissance d'un langage de programmation sera appréciée.

Programme:

Introduction : Présentation Python : caractéristiques,positionnement par rapport à d'autres langages.
Installation.Utilisation de l'interpréteur.Premier programme en Python.

Les bases : Principaux types de données : nombres, booléens, chaînes de caractères.
Déclaration de variable, typage dynamique, mots clés réservés.
Les opérateurs : priorité, associativité,opérateurs d'affectation, logiques, de comparaison.
Quelques fonctions utiles : print(), input().
Structures conditionnelles : if et elif.
Boucle while, mots clés break et continue.

Chaînes de caractères et listes : Définition et manipulation de chaînes de caractères.
Le type séquence.Les listes : définition, accès à un élément.Les références.
Les tuples.Manipulation de listes : mot clé del, fonctions list et range;
parcours d'une liste.

LY001

Le langage Python

Les fonctions	: Présentation, déclaration et appel d'une fonction. Portée des variables, mot clé global.Passage d'arguments,les arguments par défaut.La récursivité. Les fonctions Lambda : définition, utilisation. Fonctions intégrées
Programmation Objet	: Rappels sur la programmation objet.Les classes en Python.Constructeurs, attributs privés, méthodes, héritage
Les fichiers	: Méthodes d'accès aux fichiers :ouverture (accès en mode lecture, écriture, ajout, ...), fermeture, le mot clé with;lecture dans un fichier
Les types de données complexes	: Les listes de listes.Une liste de tuples.Les dictionnaires.Parcours d'un dictionnaire. Les méthodes update(), clear(), pop(), del(), values, keys();
Modules et expressions régulières	: Les modules : définition, la fonction help().Importer des fonctions Créer ses propres modules.Expressions régulières, les caractères spéciaux. Groupes et classes de caractères. Le module 're'.
Gestion des exceptions	: Principe, exemples d'exceptions.Mots clés try et except, else et finally Les assertions

Développement avancé avec Python

LY002

Durée: 3 jours

Prix et dates: nous consulter

Public:
Les développeurs en Python.

Objectifs:
Maîtriser les fonctionnalités comme la gestion des graphiques, des bases de données, les liens avec les langages C et Java, le développement d'applications Web, et l'utilisation de framework comme Django.

Connaissances préalables nécessaires:
Connaissance de base de Python.

Programme:

Programmation graphique	:	Différentes solutions : PyQt, Tkinter, PyGTK, wxWidgets, Caractéristiques de chaque solution. Travaux pratiques avec le module Tkinter.Création d'objets (fenêtres, boutons, ...), appel des méthodes associées (grid(), pack(), ...)
Gestion des bases de données	:	Les différentes méthodes :création d'une base avec les modules Gadfly,interfaçage MySQL avec MySQLdb, accès à postgreSQL avec les modules PyGreSQL ou Psycopg, ...
Développement web	:	Présentation et comparaison des frameworks et langages de template. Les frameworks disponibles :CherryPy, Paste, CPS, Django, TurboGears, Pylons, ... Les langages de templates :Myghty, Python Server Pages, Cheetah, Zope, Mise en oeuvre de Django.Installation, configuration initiale : création d'un projet, serveur de développement Django, configuration des accès aux bases de données... création et activation de modèles, développement d'une application simple.

LY002

Développement avancé avec Python

Liens avec les langages C et Java	: Les besoins : accès à des programmes en C ou à des classes Java depuis Python,bénéficier des avantages de Python depuis des programmes Java, .. Les outils :Jython, Jepp (Java Embedded Python), JPE (Java Python Extension), Boost.Python, ...

LY003

Développement web avec Django

Durée: 3 jours

Prix et dates: nous consulter

Public:
Les développeurs en Python souhaitent créer des sites web avec Django

Objectifs:
Savoir utiliser le framework Django pour le développement d'applications web

Connaissances préalables nécessaires:
Connaissance de base de Python et des concepts de base des applications web

Programme:

Développement web	: Présentation et comparaison des frameworks et langages de template. Les frameworks disponibles :CherryPy, Paste, CPS, Django, TurboGears, Pylons, ... Les langages de templates :Myghty, Python Server Pages, Cheetah, Zope
Présentation de Django	: Langage de templates système de mapping d'url design pattern MVT, principe DRY Traitement d'une requête avec le pattern MVT
Mise en oeuvre	: Installation, configuration initiale création d'un projet, configuration du projet : fichier settings.py serveur de développement Django, configuration des accès aux bases de données. authentification
Développement applicatif	: Création et activation de modèles, de vues, développement d'une application simple Les templates Requêtes de type GET et POST Utilisation des tokens CSRF pour la sécurité Déploiement des projets sur un serveur Apache

Développement web avec Django

LY003

Outils de développement	: Le framework de tests unitaires. Quelques commandes utiles : dumpdata/loaddata, dbshell, inspectdb, check... Internationalisation Fichier data Configuration Dev / Prod Création d'une commande personnalisée
API REST	: Créer un API REST avec django-rest-framework Authentification

AS120

XML Développement

Durée: 3 jours

Prix et dates: nous consulter

Public:
Développeurs, concepteurs, chefs de projet.

Objectifs:
Maîtriser kes bases du langage XML, connaître les grands principes du méta-langage.

Connaissances préalables nécessaires:
Connaissances générales des systèmes d'information.

Programme:

Introduction à XML	: Historique. Principes du langage. Concepts d'XML. Principaux domaines d'application
Règles d'écriture d'un document XML	: Structures d'un document XML. Problématiques liées à l'encodage Les espaces de noms (namespace). XHTML
Les grammaires XML	: Les DTD (Document Type Definition). Définition. Les éléments et les attributs Les schémas XML (XSD). Définition. Les éléments et les attributs. Découpage d'un schéma Les analyseurs de document XML (parseur) et les outils (XML-Spy?)
Le langage XSL	: Les feuilles de style : css, XSL Principes du langage. Le langage XPATH. Le langage XSLT. Structure d'un programme XSLT Instructions XSLT : Template, Macros, … Génération HTML depuis XML. Le langage XSL-FO.
Manipulation des documents XML avec Java	: Présentation des technologies. Les interfaces DOM (Document Object Model), et SAX (Simple API for XML)

Développement d'applications Android

UX128

Durée: 4 jours

Prix et dates: nous consulter

Public:
Développeurs et intégrateurs d'applications sous Android.

Objectifs:
Connaître les principes de fonctionnement et savoir développer des applications sous Android

Connaissances préalables nécessaires:
Connaissance de la programmation en java.

Programme:

Introduction : Présentation du système d'exploitation Android, des plate-formes matérielles, des outils :
SDK, android market, etc .. Architecture : la couche noyau Linux, l'environnement de développement, machines virtuelles, base de données SQLite
Les bibliothèques de base.

Applicatif : Les principales applications existantes. Installation/désinstallation d'une application.
Arborescence des fichiers.

Développement : Présentation du SDK pour le développement en java, installation. Utilitaires : émulateurs, simulateur de carte.
Développement d'une application de base.
Structure générale des applications. Les quatre types de composants applicatifs : Activity, Services, Broadcast receivers, Content providers.
Cycle de vie des composants.
Les versions d'android et du SDK : propriétés, adaptation aux matériels (smartphones, tablettes, télévisions, etc ...)
Les outils de développement : Android Studio IntelliJ,AVD : Android Virtual Device, ADB : Android Debug Bridge, DDMS : Dalvik Debug Monitor Server
Développement d'une première application.

UX128

Développement d'applications Android

Interface utilisateur	: Définitions. Présentation des layouts. Récupération du contexte applicatif. Les widgets. Gestion des menus. Boites de dialogue.Thèmes. Notifications (Toast, Status Bar, Dialog).
Interaction d'applications	: Présentation des Intents. Le fonctionnement des services.L'interrogation de WebServices RESTfull (client http, json) Les bonnes pratiques.
Persistance des données	: Stockage des préférences utilisateur.Le système de fichiers. Mise en oeuvre de SQLite.
Graphique et multimédia	: Développement 2D. Les APIs.Utilisation du MediaPlayer.
API	: Accès réseau, accès au système de fichiers. Capteurs internes. Gestion des périphériques : carte son, écran, caméra, clavier,.. Mise en oeuvre de la classe Sensor.

Filières Langages Java et JEE

Concepts objets et programmation Java SE 8
Programmation avancée Java SE 8

Sécurité applications Java Jee
Jee : développement web
Jee : les EJB
Développement Webservices

Serveurs d'application Jee
Administration WebSphere
Administration JBoss
Administration Tomcat

Fondamentaux programmation Java SE 8

LJ001

Durée: 5 jours
2560 €

4 au 8 mars
3 au 7 juin
7 au 11 octobre
2 au 6 décembre

Public:
Développeurs, ingénieurs logiciels et architectes d'applications.

Objectifs:
Apprendre le langage Java et assimiler les concepts objets.Utiliser les outils du JDK et les principales API de la Standard Edition 8. Tous les concepts sont illustrés par des travaux pratiques :soit sur des exemples de base, soit sur des exercices plus complets.Un projet global permettant de mettre en oeuvre l'ensemble des concepts abordés est réalisé tout au long de la formation au fur et à mesure de l'acquisition des concepts.

Connaissances préalables nécessaires:
Connaissance d'un langage de programmmation structuré

Programme:

Les concepts objet : Programmation objet, les réutilisables. Principe de l'encapsulation. Attributs et méthodes. Accesseurs. Différence entre objet et classe. Instanciation. Conventions de nommage.

Introduction à Java : Philosophie de conception sous-jacente à Java. Les différentes éditions. Présentation JSE, du jdk. Les API de la SE 8. Les fichiers sources, le byte-code et la JVM.
Présentation des différents modes d'exécution. Contrôles lors de la compilation et de l'exécution
Travaux pratiques : réalisation d'une première application.Prise en main de l'environnement de développement.

Fondamentaux programmation Java SE 8

LJ001

Syntaxe java	: Les règles d'écritures. Présentation des types primitifs, des types objets et des types abstraits. Déclaration des variables. Principaux opérateurs sur les types primitifs.Présentation des règles de priorité entre les opérateurs. Structures de contrôle : règles de définition d'une séquence d'instructions Java. Présentation des structures de contrôle conditionnelles (if-else, switch) et itératives (while, do-while, for).Tableaux : exemples de déclaration de tableau, création et initialisation. Travaux pratiques : mise en oeuvre sur des exemples simples
Les packages	: Rôle des packages. Définir ses propres packages.Travaux pratiques : création et utilisation par import d'un package applicatif.
Les classes	: Présentation des concepts orientés objet (classe, attribut, constructeur, héritage, ..)Procédures de déclaration de classes, d'attributs et de méthodes. Définition de constructeur et de l'instanciation. Travaux pratiques : exercices de prise en main et manipulation de classes Java. Mécanisme de destruction des objets : le garbage collector. Accès aux attributs et méthodes. Les références : this et null. Surcharge des noms de méthodes.Membres et méthodes de classe : static.Les classes composées d'objets.Contrôle d'accès aux membres.
Les énumérés	: Définition. Exemples.
Les interfaces	: Définition et déclaration. Utilisation des interfaces.
L'héritage	: Mécanisme d'héritage. Recherche de méthodes pour une classe dérivée. Héritage et instanciation. Conversions standards dans l'héritage. Le polymorphisme. Classes et méthodes abstraites.
Apport des Design Pattern	: Principes des solutions de conception cataloguées.Méthodologie: définition des besoins techniques, des classes "types" du pattern, des collaborations entre classes. Travaux pratiques : exemples de mise en oeuvre de patterns classiques.

Fondamentaux programmation Java SE 8

LJ001

Les classes internes	: Définition de classe interne.Caractéristiques principales. Déclaration. Exemples.
Les exceptions et erreurs	: Définition. Graphes d'héritage. Présentation du mécanisme de gestion des exceptions, des différents types d'exception Zoom sur les exceptions contrôlées.Travaux pratiques : définition d'une nouvelle exception, déclenchement et traitement de l'exception générée. Gestion des logs Java.
Les structures de données	: La classe Vector. La classe Stack. L'interface Enumeration. Structures de données ordonnées. Les collections.
Les génériques	: Définition. Exemples.
Auto Boxing et Auto UnBoxing	: Objectif. Exemples.
Les annotations	: Définition. Annotations standards. Exemples.
Accès aux SGBD	: Objectif de JDBC. Les types de drivers. Les architectures applicatives. Les classes et interfaces en jeu. Connexion. La gestion des transactions et l'isolation transactionnelle. Interrogation et mise à jour. Appel d'une procédure stockée. Les types de données. Les pools de connexion. Les Rowset. La libération des ressources. Présentation de JPA (Java Persistance API).

Programmation avancée Java SE 8

LJ002

Durée: 5 jours
2590 €

11 au 15 mars — 14 au 18 octobre
17 au 23 juin — 9 au 13 décembre

Public:
Développeurs java, ingénieurs logiciels et architectes d'applications.

Objectifs:
Approfondir la connaissance de Java notamment dans les domaines de la programmation multi-tâches, des I/O, des tests et du logging des technologies jdbc et de la persistence avec JPA.

Connaissances préalables nécessaires:
Maîtriser les concepts objets et les bases du langage Java, ou avoir suivi le stage « Concepts Objets et bases Java »

Programme:

Java SE	: Présentation des et rappels sur les classes Java. Les interfaces et les expressions lambda.
Collections	: Création de collections : ArrayList, TreeSet, HashMap, etc ... L'interface Stream. Filtrage de collections avec les expressions Lambda.
Le multi-threading	: Fonctionnement. Ordonnancement et priorité. Exclusion mutuelle. Synchronisation. Thread démon. Communication par flux "pipe".
L'API de concurrence	: Les exécuteurs de tâches. Les queues. Les maps atomiques. La représentation du temps et de ses unités. Les synchroniseurs. Les traitements asynchrones anticipés. Les variables atomiques. Les verrous "haute performance".
Les annotations	: Objectif. l'API Reflection. Annotations standards. Les méta-annotations. Fabriquer ses annotations. Annotation Processing Tool (APT)

LJ002

Programmation avancée Java SE 8

La gestion des I/O	: La gestion des flux standards. Lecture/écriture depuis la console Utilisation des streams API d'accès aux fichiers (NIO.2)
L'API JDBC	: Objectif de JDBC. Les types de drivers. Les architectures applicatives. Les classes et interfaces en jeu. Connexion. La gestion des transactions et l'isolation transactionnelle. Interrogation et mise à jour. Appel d'une procédure stockée. Les types de données. Les pools de connexion. Les Rowset. La libération des ressources. La gestion des exceptions. Présentation de JPA (Java Persistance API).
Les tests	: Objectif. Le framework JUnit.
Traçabilité des applications	: Objectif. L'API Java Logging. Gestion des dates avec l'API Java Date/Time Création de timestamps.

Sécurite Java et JEE

LJ005

Durée: 3 jours

Prix et dates: nous consulter

Public:
Tout développeur souhaitant maîtriser la sécurité des applications Java et Jee.

Objectifs:
Connaître les risques potentiels dans l'utilisation de Java, et les parades à mettre en oeuvre, les moyens de sécuriser les applications JEE.

Connaissances préalables nécessaires:
Il est demandé aux participants de connaître les notions de base du langage Java.

Programme:

Besoins	: Les risques Politique de Sécurité Evaluation des risques en fonction des différents modes d'utilisation de Java (applets, application, servlets)
Sécurisation de la JVM	: Limites naturelles imposées par Java : gestion mémoire. Contrôle du bytecode par la machine virtuelle. Mise en oeuvre du SecurityClassLoader
Protection de l'exécution	: Exécution protégée : SecurityManager, ClassLoader. Surcharge des méthodes d'accè s: lecture, écriture, exécution, ouverture de socket, autorisation de connexions... TP : Protection des accès sur le disque local d'une application.
Chiffrement	: Les mécanismes de signature. Création de clés publiques et privées. Les clés RSA, DSA. Signature d'un document. Les algorithmes SHA1withDSA, MD5withRSA. Les MessageDigest. Les algorithmes MD2, MD5, SHA-1, SHA-512 TP : Vérification de l'authenticité d'un document

LJ005

Sécurite Java et JEE

Certificats : Cycle de vie d'un certificat. La fabrique de certificats Java.
Les certificats de modification X509.

Contrôle : Rappel sur les ACL. Le paquetage java.security.acl. Ajout d'entrée, vérification d'accès.

Obfuscation : Principe
Techniques d'obfuscation
Solutions commerciales

JAAS : Présentation
Fonctionnement et mise en oeuvre

Sécurité Jee : Exemples avec WebSphere et JBoss
Le service de sécurité
Sécurité Web et EJB
Autorisations EJB V3
Accès applicatifs et lien avec un annuaire Ldap
Mise en oeuvre des certificats avec JEE.

JEE : Développement d'applications web

AS004

Durée: 3 jours

Prix et dates: nous consulter

Public:

Les développeurs java soohaitant intégrer les technologies des servlets et des pages JSP.

Objectifs:

Mettre en place une application web dynamique à l'aide de servlets, JSP, Taglibs et des JavaBeans en respectant le modèle MVC. Assurer la persistance en utilisant JDBC.

Connaissances préalables nécessaires:

Il est demandé aux participants de connaître la programmation Java, ainsi que les techniques de base Internet (HTML, serveur HTTP).

Programme:

L'API JDBC : Objectif de JDBC. Les types de drivers. Les architectures applicatives.
Les classes et interfaces en jeu. Connexion.
La gestion des transactions et l'isolation transactionnelle. Interrogation et mise à jour. Appel d'une procédure stockée. Les types de données. Les pools de connexion. Les Rowset. La libération des ressources. La gestion des exceptions.
Présentation de JPA (Java Persistance API).

Architecture en couches : Présentation, Métier et Persistance. Couplage fort, couplage faible. Les Design Patterns nécessaires : Singleton, Factory, Façade, Iterator.Notion de composant. Découpage du composant en 3 couches (service, donnée, persistance). Mapping opérationnel pour la persistance. L'implémentation de la persistance avec JDBC.

La couche Présentation : Servlet, JSP et Taglib. Design Pattern MVC

AS004

JEE : Développement d'applications web

Servlet (le contrôleur)	: Objectif. Le procotole HTTP. L'API Servlet. Cycle de vie d'une servlet. Gestion de contexte. Gestion de la requête client. Gestion des cookies. Redirection côté client et côté serveur. Configuration et déploiement.
Java Server Page (la vue)	: Objectif. Cycle de vie d'une page JSP. JSP dans le MVC. Les différents tags. JSP et l'intégration des JavaBeans.
Les bibliothèques de Tags et JNDI	: Objectifs. Utilisation et conception. La JSTL. Objectif de JNDI. Enregistrement (Binding. Lookup)

JEE : les EJB

AS005

Durée: 3 jours

Prix et dates: nous consulter

Public:
Développeurs Java, concepteurs, chefs de projet.

Objectifs:
Mettre en oeuvre des applications Jee manipulant des EJB 3. Développer des EJB3. Déployer les applications Jee dans un serveur d'applications.

Connaissances préalables nécessaires:
Il est demandé aux participants de connaître la programmation en Java. La connaissance des architectures distribuées est un plus.

Programme:

Les concepts des architectures distribuées	:	L'architecture distribuée : C/S, architecture n-tiers. L'architecture serveur d'applications. La plateforme Java EE. Le RMI (Remote Method Invocation).
Programmation avec JNDI	:	Présentation de Java Naming and Directory Interface. Les services de désignation, d'annuaire. L'architecture JNDI.
Les spécifications de Java EE 5	:	Les annotations Les génériques
Les spécifications EJB2 et EJB3	:	Normes EJB 2.0, EJB 3.0. Le Modèle Vue Controleur (MVC). Contexte d'utilisation des EJB. Les différents types d'EJB : session, entity, message driven.
Les EJB session stateless, stateful	:	Utilité, cycle de vie, développement, déploiement. Mode conversationnel avec les beans à état. Problématiques de concurrence et de clustering. EJB session et Web Services.

AS005

JEE : les EJB

Les EJB entité et la norme JPA	: Norme JPA (Java Persistence API) et configuration. Gestionnaire de persistance. Relation avec les graphes d'objets. Les mécanismes d'héritage.
Le langage EJB-QL (Query Language)	: Les types de requêtes. Jointure et restrictions.
Gestion des transactions	: Les différents modèles transactionnels. Gestion des transactions distribuées. Transactions de niveau conteneur ou bean. Transactions au niveau du client.
Gestion de la sécurité	: Sécurité Jee avec JAAS. Sécurité par programmation. Sécurité déclarative.
Architecture MOM avec les EJB MDB	: Rappel des concepts JMS et MOM. Développement d'EJB MDB (message driven bean) et de clients. Déploiement d'une architecture MOM.

Développement Web Services

AS006

Durée: 3 jours

Prix et dates: nous consulter

Public:
Les chefs de projets et développeurs souhaitant concevoir et développer des web services.

Objectifs:
Savoir développer des applications utilisant les techniques des Web services.

Connaissances préalables nécessaires:
Ce cours présente les WebServices en environnement Java.

Programme:

Introduction	: Historique. Définitions. Les différents types de webservices : ws-* et RESTful. Les EJB3 et annotations Java pour créer des webservices.
Le protocole SOAP	: Présentation : Simple Object Access Protocol pour l'échange de messages XML. L'interopérabilité avec SOAP, les avantages. Structure d'un message.Exemples.
WSDL	: Définition. Structure d'un document WSDL. Définition d'un service. Gestion de la sécurité.
Les annuaires UDDI	: Universal Description, discovery and Integration pour la recherche des services web disponibles. Les annuaires publics. Structure des données. Mise en oeuvre de jUDDI. Publication d'un WebService.
Les API Java pour XML	: JAXP, JAXB, SAAJ : pour le traitement des données XML JAX RPC, lancement de procédures distantes JAXM, messages XML; JAXR, identification de services web Le WSDP : Java Web Services Developper Pack.

Développement Web Services

AS006

WebServices et Axis	: Présentation Axis. Principe de fonctionnement. Mise en oeuvre. Maintien de session avec la méthode setMaintainSession() Gestion des attachements. Sérialisation personnalisée avec typeMapping.
Axis 2	: Présentation, historique Fonctionnalités. Travaux pratiques : installation, lancement du serveur, tests.
Web Services et sécurité	: Le besoin. Identification des menaces. Différents moyens de sécurisation : WS-Security, Username Token, X.509 Certificate Token Profile. L'authentification HTTP. Authentification du client. Création des rôles de sécurité. Pose de contraintes. Vérification. Génération de clés. Déclaration du connecteur sécurisé sous Tomcat.
Le standard JAX-RS	: La technologie des webservices RESTful : Representational State Transfer. Format des données transférées : XML, JSON. Le WADL : Web Application Description Language. Les implémentations : Apache-CXF, Jersey. Mise en oeuvre de webservices RESTful avec Jersey.
EJB3	: Apports des EJB3. Génération de webservices à l'aide des annotations.

Serveurs d'application JEE

AS011

Durée: 2 jours

Prix et dates: nous consulter

Public:
Les chefs de projets et toute personne souhaitant comprendre le fonctionnement de l'architecture JEE.

Objectifs:
Connaître l'architecture et les principes des serveurs d'applications. Savoir concevoir une application avec le modèle JEE.

Connaissances préalables nécessaires:
Aucune connaissance préalable n'est requise pour suivre ce cours.

Programme:

Serveurs d'application	: Introduction. Transactions. Architecture des applications web. Les différents éléments et leurs rôles
Le modèle JEE	: L'architecture JEE. Le modèle JEE. servlets, Java Server Pages, EJB. Spécifications. Les composants d'un serveur d'application Java.
Les produits	: Présentation de différents serveurs d'application du marché : Geronimo, JBoss, Jonas, WebLogic, WebSphere Comparatif produits : version de JDK, type d'administration, automatisation, industrialisation, besoin en ressources, support des EJB3, ... Portabilité des applications JEE.
Administration	: Définition des différents objets à gérer : serveurs Web, serveur d'application, moteur de servlets, container, EJB, hôtes virtuels, connecteurs JDBC. Le service de nommage JNDI.

Serveurs d'application JEE

AS011

Conteneur Web	: Servlets, pages JSP : pages HTML dynamiques, communication avec bases de données et applications Java. Les frameworks de développement : objectifs et techniques mise en oeuvre Struts, JSF, Apache MyFaces, Spring.
Développement avec les EJB	: Les EJB : spécifications (état actuel et limites). EJB entité, EJB session, EJB Message-driven. Apports des EJB3. Jointures. Transactions. Présentation de EJB-QL. Relations entre le développeur d'EJB et le DBA.
Intégration	: Transactions. Utilisation de JTA, JTS. Transactions explicites. Transactions gérées par container. Sécurité : JAAS, rôles, groupes de permissions. Mécanismes de connexion aux bases de données : JDBC, pool de connexions. Connexion interapplicatives avec JCA.
Déploiement et exploitation	: Cycle de vie d'une application. Industrialisation. Les fonctions à assurer : code, assemblage, nommage, création des fichiers de description en XML, des fichiers jar, mise en production. Les outils de développement: eclipse, WSAD. Les outils de mise en production.

Administration WebSphere

AS131

Durée: 5 jours

Prix et dates: nous consulter

Public:
Techniciens d'exploitation, administrateurs WebSphere.

Objectifs:
Connaître l'architecture et les principes du serveur d'applications WebSphere. Savoir installer, configurer et exploiter des applications sous WebSphere. Les travaux pratiques sont réalisés avec la version 7 et 8.

Connaissances préalables nécessaires:
Une bonne connaissance des concepts de l'internet, des systèmes d'exploitation, et quelques notions de base sur les bases de données.

Programme:

WebSphere : Présentation des fonctionnalités du produit WebSphere.
Rappels sur la terminologie et les objets WebSphere Application Server. Architecture. Topologie WebSphere.
Les nouveautés de la version WebSphere Application Server V8.

Installation : Le produit WebSphere Application Server, Base de données, IBM-HTTP server, jdk.
Installation manuelle. Installation automatique.

Administration du système : Fonctionnement de la console
Les objets à administrer : arborescence, groupes de serveurs, noeuds, applications d'entreprises.
Paramétres de configuration : la base de données de configuration, les fichiers XML.

Intégration : Utilisation de l'outil d'assemblage d'applications (WRD).
Le service de nommage : l'interface JNDI.
Gestion des hôtes virtuels
Variables WebSphere.
Domaines de réplication
Création de profils serveurs.

Administration WebSphere

AS131

Ressources	: Sources de données. Exemple: mise en oeuvre avec PostgreSQL. Modification de la base de données cible. Fournisseurs d'URL, fournisseurs JMS, les sessions JavaMail. Connexion interapplicatives avec JCA. Gestion du pool de connexions sous WebSphere Application Server.
Sécurité	: Les niveaux de sécurité dans WebSphere (Système, ressources, les rôles, etc ...). Définition des rôles. Authentification ldap. Configurer la sécurité du serveur
Outils d'administration	: Console d'administration. scripts prédéfinis Outils de migrations, collector. Administration à distance, automatisation. wsadmin : présentation, objets supportés : $AdminApp, $AdminConfig, $AdminControl, $AdminTask JACL: Syntaxe de base, contrôle de programme, exemples. Jython: Syntaxe de base, contrôle de programme, exemples.
Surveillance	: Informations émises par WebSphere, les exceptions, les messages de la console d'administration, les fichiers de traces. Journalisation. Journal de la JVM. Vérification de la configuration. Résolution des incidents
Gestion des performances	: Etude du Ressource Analyser. Exploitation de l'advisor.
Multi serveurs	: Déclaration de plusieurs serveurs sur la même machine dans le même profil. Routeur HTTP IBM : Configuration Mise en place d'un cluster avec affinités de Session sans gestionnaire de déploiement

Administration Tomcat

AS133

Durée: 3 jours

Prix et dates: nous consulter

Public:
Toute personne souhaitant configurer et administrer une application avec Tomcat.

Objectifs:
Comprendre le fonctionnement de Tomcat, et savoir le mettre en oeuvre, l'installer, le configurer et l'administrer, optimiser le fonctionnement du serveur. Ce module s'appuie sur des travaux pratiques.

Connaissances préalables nécessaires:
Il est demandé aux participants de connaître les bases tcp/ip, http, java (jsp, servlets)

Programme:

Concepts de base	:	L'architecture des applications web : les différents composants. Présentation de Tomcat. Les versions de Tomcat. Utilisation de Tomcat avec le serveur Web Apache Servlets, JSP et composants, architecture d'un site Java
Installation	:	Installation de Tomcat et modification des paramétres de base. Exercice : installation du serveur Tomcat, positionnement des variables d'environnement, lancement du serveur, tests de fonctionnement.
Configuration	:	Architecture de Tomcat : "engine", services, "context". L'arborescence, les variables d'environnement : JAVA_HOME, TOMCAT_HOME, CLASSPATH.
Console Manager	:	Présentation, fonctions disponibles : état du serveur, accès à la documentation, à la console d'administration.

Administration Tomcat

AS133

Déploiement d'applications web	: Descripteurs XML, les fichiers .war. Déploiement à chaud.Tomcat Manager. Travaux pratiques : déploiement d'une application simple par l'interface d'administration de Tomcat.
Console d'administration	: Travaux pratiques : installation et configuration des droits d'accès dans le fichier tomcat-users.xml Fonctions disponibles : liste des ports écoutés par tomcat pour les requêtes http, applications déployées sur chaque hôte virtuel, configuration des connecteurs, de la sécurité, (utilisateurs, groupes, rôles), des sources de données.
Sécurité	: Sécurisation et permissions. Les domaines, les rôles : définitions, principe de fonctionnement et configuration. Gestion des utilisateurs, modification de rôles, gestion des mots de passe. Le stockage des informations de sécurité : JDBC, Datasource, JNDI, JAAS, mémoire Security Manager. Travaux pratiques : configuration de domaines de sécurité, création et affectation d'utilisateurs, spécification des pages protégées, mise en place des mots de passe
Fichiers de configuration	: Etude des paramétres à positionner dans le fichier server.xml, les balises server, engine, host, context, logger, loader. Le descripteur de déploiement web.xml : déclaration de la servlet, lien entre la servlet et la requête. Configuration des sources de données : présence des pilotes, optimisation des paramétres de connexion. Travaux pratiques : mise en place d'un pilote pour l'accès à une base postgreSQL. Les connecteurs : la balise connector. Le connecteur HTTP, fonctionnement Tomcat en standalone. Sécurisation : configuration du protocole https.

AS133

Administration Tomcat

Performances et tests	: gestion de la charge : load-balancing. Mise en place de clusters. Interface apache/tomcat avec mod-jk. Travaux pratiques : installation et configuration d'un répartiteur mod_jk Conservation des sessions : mise en place des sticky sessions.
Journalisation	: Mise en oeuvre de la journalisation avec log4J. Analyse des logs. Fichiers de logs, résolution d'incidents. Travaux pratiques : configuration de Log4j. Modification des niveaux de journalisation. Différents types ventilations. Centralisation des logs vers une machine syslog externe.
Exploitation	: Intégration JMX. Suivi des performances. Suivi du ramasse miettes (garbage collector) avec jconsole. Paramétres de la JVM pour gérer au mieux l'espace mémoire. Travaux pratiques : mise sous stress avec jmeter, suivi et amélioration de la configuration

Administration JBoss

AS134

Durée: 3 jours

Prix et dates: nous consulter

Public:
Exploitants, administrateurs d'applications JEE fonctionnant avec JBoss.

Objectifs:
Savoir installer et configurer JBoss. Savoir intégrer une application JEE, en assurer la mise en production, l'exploitation.

Connaissances préalables nécessaires:
Une bonne connaissance des concepts de l'internet, des systèmes d'exploitation, et quelques notions de base sur les bases de données.

Programme:

JBoss : JBoss présentation
Historique, présentation des différentes versions et de leurs caractéristiques, de JBOss AS V 5, 6, 7 à WildFly.
L'architecture Jboss : le noyau, la couche services, la couche présentation, la couche application, le principe JMX et les Mbeans.
Les outils JBoss: Hibernate, AOP, cache IDE. Présentation du noyau JBoss. Présentation du micro-conteneur.
Installation, configuration, arborescence des fichiers.Les services disponibles.
Adaptation de la configuration : ajout/suppression de services
Visualisation dans les consoles de JBoss.

Applicatif : Terminologie: application web, container, sources de données, pilote JDBC, hôtes virtuels.
Configuration des services essentiels : JBossWeb, JNDI, JBossMQ, ...
Configuration du scanner de déploiement d'applications.

Administration JBoss

AS134

Lien avec Tomcat	: Installation de Tomcat et modification des paramétres de base Activation du service Tomcat dans la configuration Jboss. Le fichier server.xml. Administration du service web. Mise en place d'hôtes virtuels.
Gestion des ressources	: Fournisseurs d'URL, fournisseurs JMS, sessionMail. Mécanismes de connexion aux bases de données : JDBC, pool de connexions. Modification de la base de données cible. Exemple avec postgresql.Connexion interapplicatives avec JCA.
Sécurité avec JAAS et SecurityManager	: Intégration de la sécurité dans le serveur d'applications. Mise en place d'une politique de sécurité lors du déploiement de l'application. Sécurité: JAAS, rôles, groupes de permissions.
Automatisation	: Exploitation par scripts avec twiddle, JBoss-cli. Parcours de ressources. Ecriture d'un script de visualisation de statistiques sur un cluster: taux d'utilisation mémoire, nombres de requêtes, connexions JDBC. Scripts d'automatisation.
Journalisation	: Mise en oeuvre de la journalisation. Analyse des logs Fichiers de logs, résolution d'incidents. Configuration de la journalisation. Modification des niveaux de journalisation. Différents types ventilations. Centralisation des logs vers une machine syslog externe.
Suivi	: Gestion des performances. Suivi du garbage collector. Installation du MBean Jboss Profiler. Suivi du ramasse miettes (garbage collector) avec jconsole. Paramétres de la JVM pour gérer au mieux l'espace mémoire. Travaux pratiques: mise sous stress avec jmeter, suivi de la charge et amélioration de la configuration. Optimisation et allègement des configurations JBoss.

Conditions de vente

Tous les prix indiqués dans ce catalogue sont hors taxes.

Toute inscription à une formation implique l'adhésion des présentes conditions de vente.
L'inscription est ferme à partir de la signature de la convention. Pythagore F.D. se réserve la possibilité d'annuler une session si le nombre de stagiaires est insuffisant. Pythagore F.D. informera le client au minimum dix jours ouvrables avant le début du stage. Le client peut alors reporter ou annuler son inscription.

Annulations

Toute formation commencée est intégralement due. Si l'annulation d'une inscription à un stage inter-entreprise, est faite dans la période allant du 10è jour ouvré au 2è jour ouvré avant le début du stage , 50% du montant des frais de formation sont dus. Si l'annulation n'a pas été faite 48h avant le début de la formation, la totalité du montant des frais de formation reste due. Dans le cas de l'annulation d'une formation ou d'un cycle de formation en intra moins de dix jours ouvrables avant le début de la formation, 50% des frais de formation restent dus.

Déroulement du stage

Tout stagiaire doit se conformer au règlement du centre de formation de Pythagore F.D. A défaut, le stagiaire pourra être exclus de la formation. Les frais de formation sont alors intégralement dus. Pendant la formation, les stagiaires restent les employés du client qui est responsable de leurs faits et gestes en application de l'article 1384 du code civil.

Paiement

Les factures sont payables , sans escompte, à réception pour les stages en inter-entreprises. Les cycles de formation d'une durée supérieure à un mois font l'objet d'une facturation mensuelle. En cas de non-paiement à son échéance, toute somme due portera intéret de plein droit au taux d'une fois et demie le taux d'intéret légal.

Litiges

Tout litige qui ne pourra être réglé à l'amiable sera du ressort du tribunal de Commerce de Paris.

Pythagore F.D. - 11, rue du Faubourg Poissonnière 75009 PARIS
Tél : 01 55 33 52 10 - Fax : 01 55 33 52 11
S.A.S au capital de 40 000 Euros - RCS Paris B 398 145 474

Pour toute information, appelez-nous au
01 55 33 52 10
www.Pythagore-fd.fr

Table des matières