Cours de Java

Sciences-U Lyon

Java - Introduction

Java - Fondamentaux

Java – Avancé

http://www.rzo.free.fr

Sommaire

- Java – Introduction
- Java – Fondamentaux
  - **Histoire de Java**
  - Machine Virtuelle
  - Documentation
  - Qualité logicielle
  - ...

Sommaire

- Java – Fondamentaux
  - ...
  - Rappel d'algorithmie
  - Structure de programme
  - Langage Java
  - Performances de Java
  - Conception
- Java - Avancé

Histoire de Java

- **1980's :** Bill Joy tente de réécrire Unix, se heurte à la complexité du C++
-
- **1991 :** Green Project, Sun - langage pour appareils électroménagers (fiable, peu couteux, simple)
-
- **1991 :** développement de C++ en C++ ++ --, puis Oak
-
- **1994 :** LiveOakSystem, Système d'Exploitation basé sur Java
-
- **1994 :** HotJava, browser supportant les applets
-
- **1995 :** présentation officielle à SunWorld 95.

Histoire de Java

- Facteurs de Succès
  - Gratuité
  - Synchrone avec développement d'Internet
- Ce qu'est Java
  - Langage de programmation
  - Machine virtuelle

Sommaire

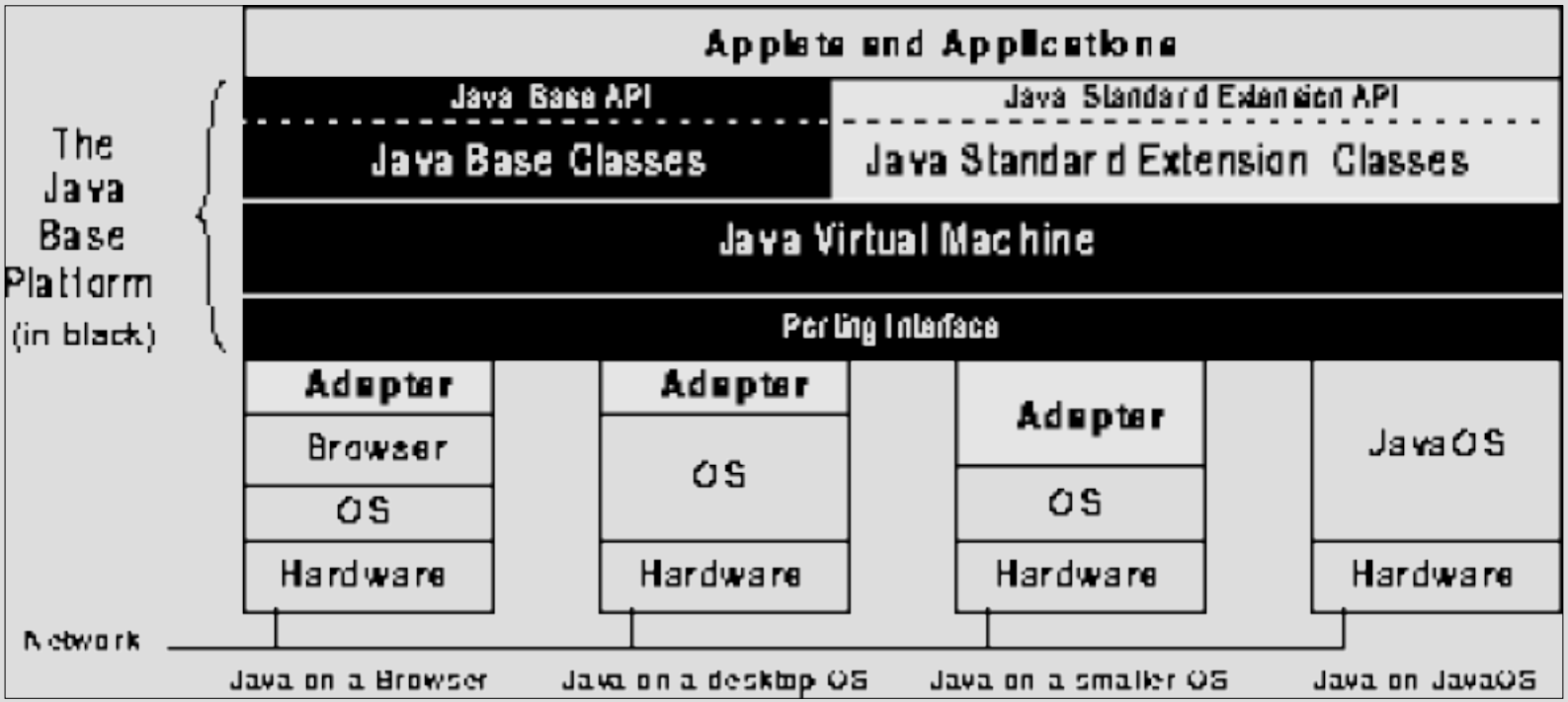
- Java – Introduction
- Java – Fondamentaux
  - Histoire de Java
  - **Machine Virtuelle**
    - **Implémentation**
    - **Du code au programme**
    - **API d'extensions standard**
  - Documentation
  - ...

Sommaire

- Java – Fondamentaux
  - ...
  - Qualité logicielle
  - Rappel d'algorithmie
  - Structure de programme
  - Langage Java
  - Performances de Java
  - Conception

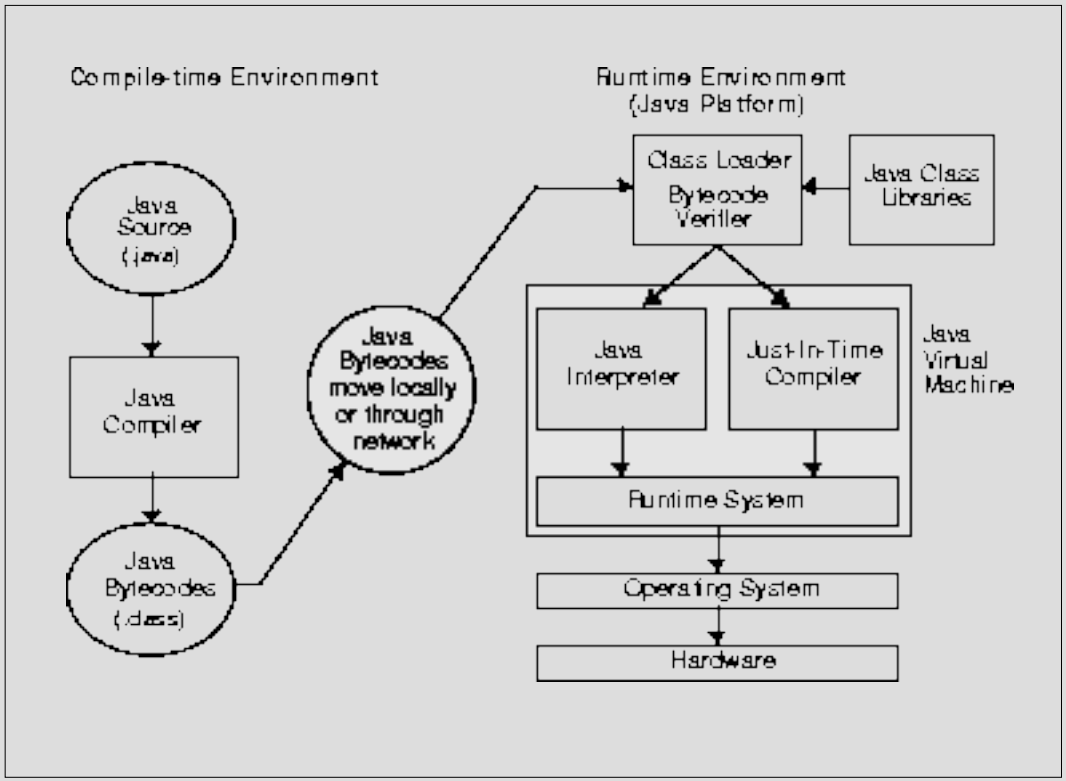
Machine Virtuelle

- Implémentation
  - Software ou Hardware



Machine Virtuelle

- Du code au programme



Machine Virtuelle

- APIs d'extension standard
  - Java 3D, Video, MIDI,
  - Java Share,
  - Java Telephony,
  - Java Server
  - Java Management.

Sommaire

- Java – Introduction
- Java – Fondamentaux
  - Histoire de Java
  - Machine Virtuelle
  - **Documentation**
  - Qualité logicielle
  - ...

Sommaire

- Java – Fondamentaux
  - ...
  - Rappel d'algorithmie
  - Structure de programme
  - Langage Java
  - Performances de Java
  - Conception
- Java - Avancé

Documentation

- Conception, implémentation
  - Problématique
  - Cahier des charges
    - Fonctionnalités
  - Conception globale
    - Analyse, choix de la solution, segmentation
  - Conception des sous-parties
    - Analyse, choix de la solution, tests

Documentation

- Conception, implémentation
  - Intégration
  - Tests
    - Boite noire
    - Boite blanche
  - Validation
    - Selon le cahier des charges
  - Perspective du sujet

Sommaire

- Java – Introduction
- Java – Fondamentaux
  - Histoire de Java
  - Machine Virtuelle
  - Documentation
  - **Qualité logicielle**
  - ...

Sommaire

- Java – Fondamentaux
  - ...
  - Rappel d'algorithmie
  - Structure de programme
  - Langage Java
  - Performances de Java
  - Conception
- Java - Avancé

Qualité logicielle

- Objectifs
  - ✭ Les programmes doivent être :
  - Efficients
    - réalisation des tâches dont l'utilisateur a besoin
  - Efficaces
    - réalisation des tâches de manière rapide et performante.
  - Intuitifs
    - les tâches courantes doivent pouvoir se faire sans documentation ni formation.

Qualité logicielle

- Utilisateurs non informaticiens
  - Manipulations complexes à proscrire
  - Bugs très mal percus
  - Un logiciel doit être une aide, pas une charge

Qualité logicielle

- Validation des méthodes
  - Écriture des tests avant les méthodes
  - Prise en compte des cas particuliers
- Documentation du code
  - Voir Javadoc
- Méthode des assertions
  - Tests de validité des paramètres dans le code

Qualité logicielle

- Framework de tests unitaires
  - Débuggage : de 50% du temps (Expert), à 90 % du temps (débutant)
- Exemple : Junit

Qualité logicielle

- Exemple : Junit

```
package tests;

public class HelloWorldTest {
    protected String msg;


    public HelloWorldTest(String message) {
this.msg = message;
    }

    public String act(){
System.out.println(this.msg);
return this.msg;
}

    public static void main(String[] args){
HelloWorldTest HWT = new HelloWorldTest("HelloWorld !");
String s = HWT.act();
    }

}
```

Qualité logicielle

- Exemple : Junit

```
package tests;

public class TestHelloWorldTest extends junit.framework.TestCase {

    public void TestHelloWorldTest {
String s = "Bonjour Monde !";
HelloWorldTest HWT = new HelloWorldTest(s);
Assert.assertTrue(HWT.act().equals(s));
    }

    public void main(String[] args) {
junit.textui.TestRunner.run(TestHelloWorldTest.class);
    }
}
```

Qualité logicielle

- Xtreme Programming
  - http ://www.extremeprogramming.org/
  - Méthodologie
  - Objectif : satisfaction du client
  - Communication, Simplicité, feedback, courage
  - Règles simples

Qualité logicielle

- Xtreme Programming : Planning
  - Annoncer un planning créé un planning
  - Editer régulièrement des versions intermédiaires
  - Mesurer l'avance du projet
  - Division du projet en itérations
  - créer un planning pour chaque itération
  - Echange quotidien entre les développeurs
  - Revenir à l'XP quand la méthodologie se relache

Qualité logicielle

- Xtreme Programming : Design
  - Simplicité
  - Utiliser des métaphores
  - Introduire les fonctionnalités le plus tard possible
  - Réécrire le code aussi souvent que possible

Qualité logicielle

- Xtreme Programming : Codage
  - En fonction des besoins du client
  - Respecter les standards
  - Coder les tests en premier
  - Coder en binôme
  - Intégration du programme par un seul binôme

Qualité logicielle

- Xtreme Programming : Codage
  - Intégration aussi souvent que possible
  - Le code de tous appartient à tous
  - Optimiser le plus tard possible
  - Pas d'heures supplémentaire

Qualité logicielle

- Xtreme Programming : Test
  - Tout le code doit avoir son unité de test
  - Tout le code doit être testé avant d'être publié
  - Créer des tests pour chaque bug
  - Réaliser des tests d'acceptation aussi souvent que possible (tests utilisateur)

Sommaire

- Java – Introduction
- Java – Fondamentaux
  - Histoire de Java
  - Machine Virtuelle
  - Documentation
  - Qualité logicielle
  - ...

Sommaire

- Java – Fondamentaux
  - ...
  - **Rappel d'algorithmie**
  - Structure de programme
  - Langage Java
  - Performances de Java
  - Conception
- Java - Avancé

Rappel d'algorithmie

- Les boucles
  - while(condition){action}
  - do{action}while()
  - for(i=0;i<5;i++){action}
- TP : implémentation d'un algorithme de Tri
  - Conception : présentation des différents algorithmes de tri

Sommaire

- Java – Introduction
- Java – Fondamentaux
  - Histoire de Java
  - Machine Virtuelle
  - Documentation
  - Qualité logicielle
  - ...

Sommaire

- Java – Fondamentaux
  - ...
  - Rappel d'algorithmie
  - **Structure de programme**
  - Langage Java
  - Performances de Java
  - Conception
- Java - Avancé

Structure de programme

- Diagramme de Classe UML
  - Package, Classe, attribut, constructeur, new, méthode, variable locale, héritage
- Diagramme de Séquence UML
  - Instance, message, déclaration, initialisation, instanciation

Java - Fondamentaux

- Bilan