



Интерфейси и вътрешни класове в Java

Лектор:
Инж. Божидар Бацов

Java Workshop

- Адрес - бул. Джеймс Баучер 5, ФМИ, зала 100
- Начало 31.10.2009(Събота), 15:00 часа
- Тъй като местата са ограничени е хубаво, който има лаптоп да си го носи
- **НИКАКВА ТЕОРИЯ, САМО ПРАКТИКА!**

Проектът Spellbook

- <http://code.google.com/p/spellbook-dictionary/>
- Работа по истински ООП проект на Java
- Различни отговорности – писане на код, документация, тестване

Сравняване на обекти за идентичност

- Метода equals() на класа Object
- Правилно имплементиране на equals()
 - Проверка за сравнение с null
 - Проверка за сравнение със същия обект
 - Проверка на типа на сравнявания обект
 - Сравняване на двата обекта по полета
 - Това отново може да използва equals()
- Метода hashCode()

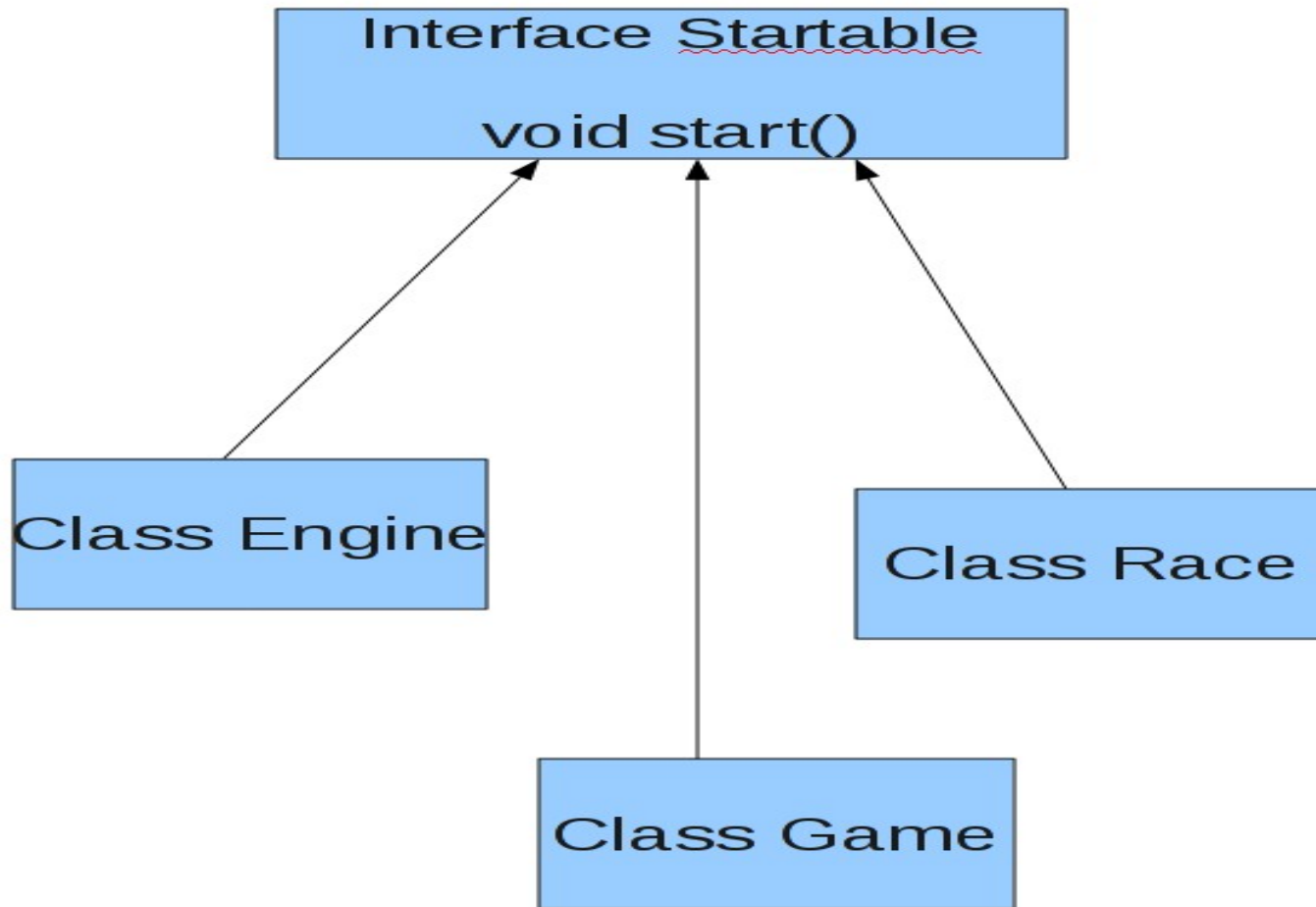
Пример в NetBeans

Преглед на задачите за ДОМАШНО

- Решенията са достъпни в kenai.com
- Ключови моменти в двете задачи
- Често срещани грешки

Интерфейс

- Езикова конструкция подобна на клас, която съдържа контракт(набор от задължения), които един клас трябва да изпълни
- Наподобява напълно абстрактен клас
- Обикновено съдържа само методи
- Всички методи имат ниво на достъп `public` и са абстрактни



Интерфейс

```
Interface InterfaceName {  
    Fields;  
    ....  
    Methods;  
}
```

Имплементиране на интерфейс

```
class aClass implements aInterface{  
    // class definition  
    // implements interface methods  
}
```

Интерфейси

- Всички методи в тях са абстрактни
- Един клас може да имплементира повече от един интерфейс
- Интерфейса НЕ Е КЛАС – обекти от него не могат да бъдат създавани
- Могат да бъдат деклариране променливи от интерфейсен тип
- Работят с instanceof оператора

Интерфейси

- Абстрактен клас може да имплементира интерфейси(напълно или частично) – това всъщност е популярна практика
- Интерфейс -> Абстрактен Клас -> Конкретен клас
- Преглед на няколко интерфейса от стандартната библиотека

Интерфейсът Comparable

```
public interface Comparable<T> {  
    int compareTo(T other);  
}
```

Пример в NetBeans

Някои особености

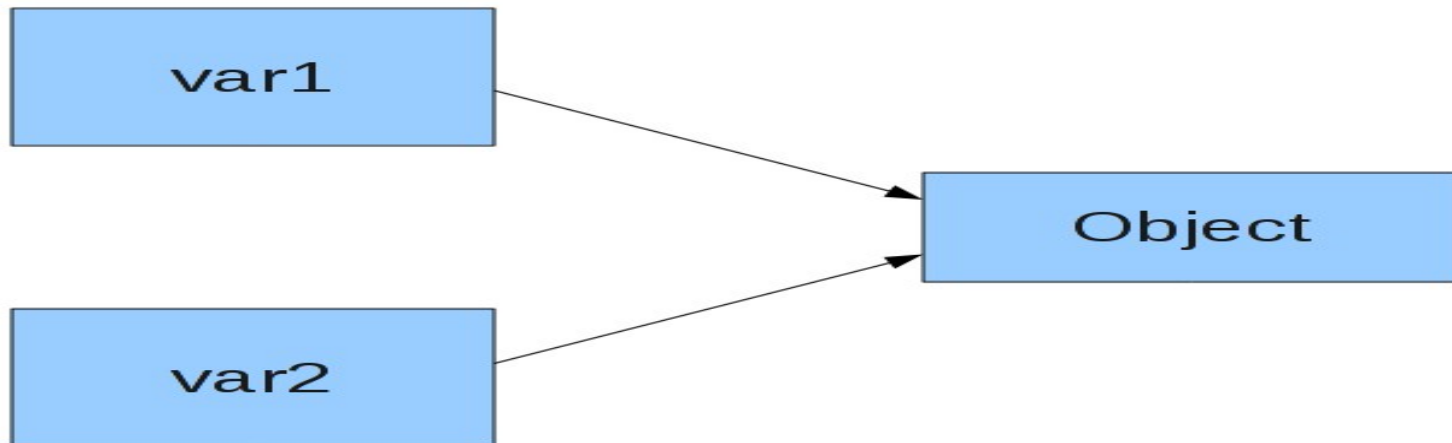
- Не могат да съдържат instance полета
- Не могат да съдържат статични методи
- Всички полета в интерфейс са константи (public static final)
- Един интерфейс може да наследди(разшири) друг
- Интерфейси маркери – нямат методи

Клониране на обекти

- Метода `clone()` на класа `Object`
- Интерфейсът `Cloneable` – не съдържа методи, интерфейс маркер(`tag`)
- Плитко клониране
- Дълбоко клониране
- Имплементиране на `clone()`
- Клониране на масиви

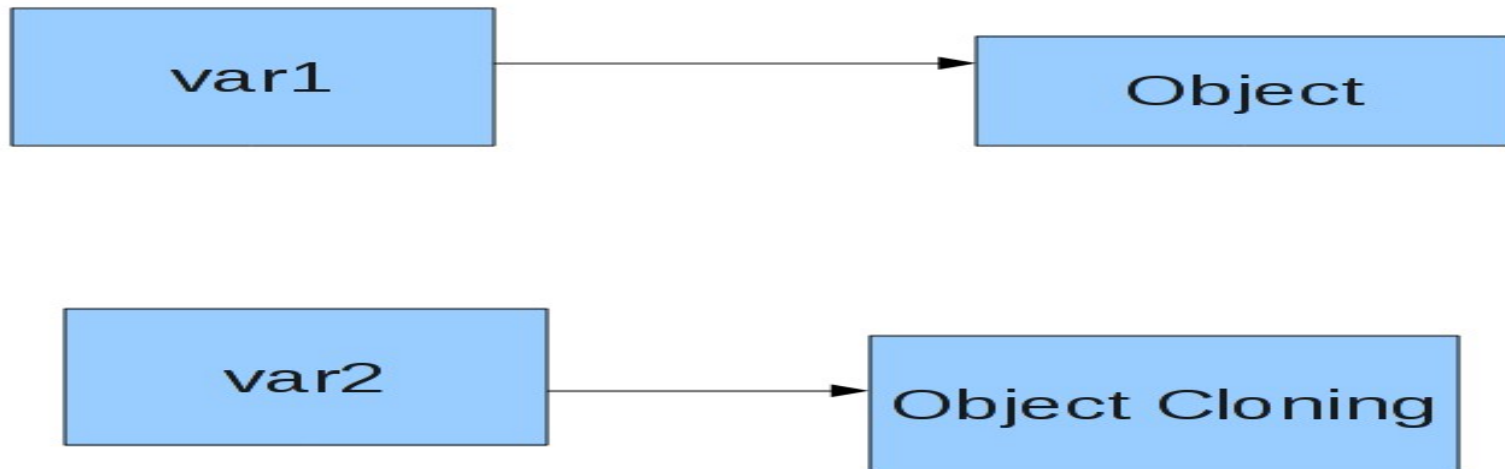
```
SomeClass var1 = new SomeClass()
```

```
SomeClass var2 = var1
```



```
SomeClass var1 = new SomeClass()
```

```
SomeClass var2 = var1.clone()
```



Пример в NetBeans

Употреба на интерфейси за callback

- Callback е код извикан в резултат на настъпването на някакво събитие – изтекъл таймер, натиснат бутон от клавиатура и т.н.
- Реализиране се посредством интерфейс с action методи
- Класът Timer

Пример в NetBeans

Вътрешни класове

- Представяват влагането на дефиницията на един клас в друг
- Реализирани са на ниво компилатор
- Позволяват достъп до вътрешното състояние на класа, в когото са вложение
- Видове – стандартни, локални, анонимни и статични

Стандартни вътрешни класове

- Дефинирани са в тялото на друг клас на ниво полета и методи
- Имат достъп до полетата и методите на вътрешния клас
- Имат скрита референция към външния клас

```
class OuterClass {  
    Fields;  
    Methods;
```

```
    class InnerClass {  
        Fields;  
        Methods;  
    } // end of inner class  
} //end of outer class
```

Пример в NetBeans

Локални вътрешни класове

- Дефинирани са в тялото на метод на клас
- Имат достъп до локалните променливи в метода
- Не са видими извън метода, в който са дефинирани

Пример в NetBeans

Анонимни вътрешни класове

- Разновидност на локалните класове
- Създава се обект от клас дефиниран след оператора за присвояване
- Обикновено анонимия клас е имплементация на някой интерфейс

Пример в NetBeans

Статични вътрешни класове

- Еквивалентни на стандартните вътрешни класове, но без референция до външния клас
- Служат като допълнително пространство на имена