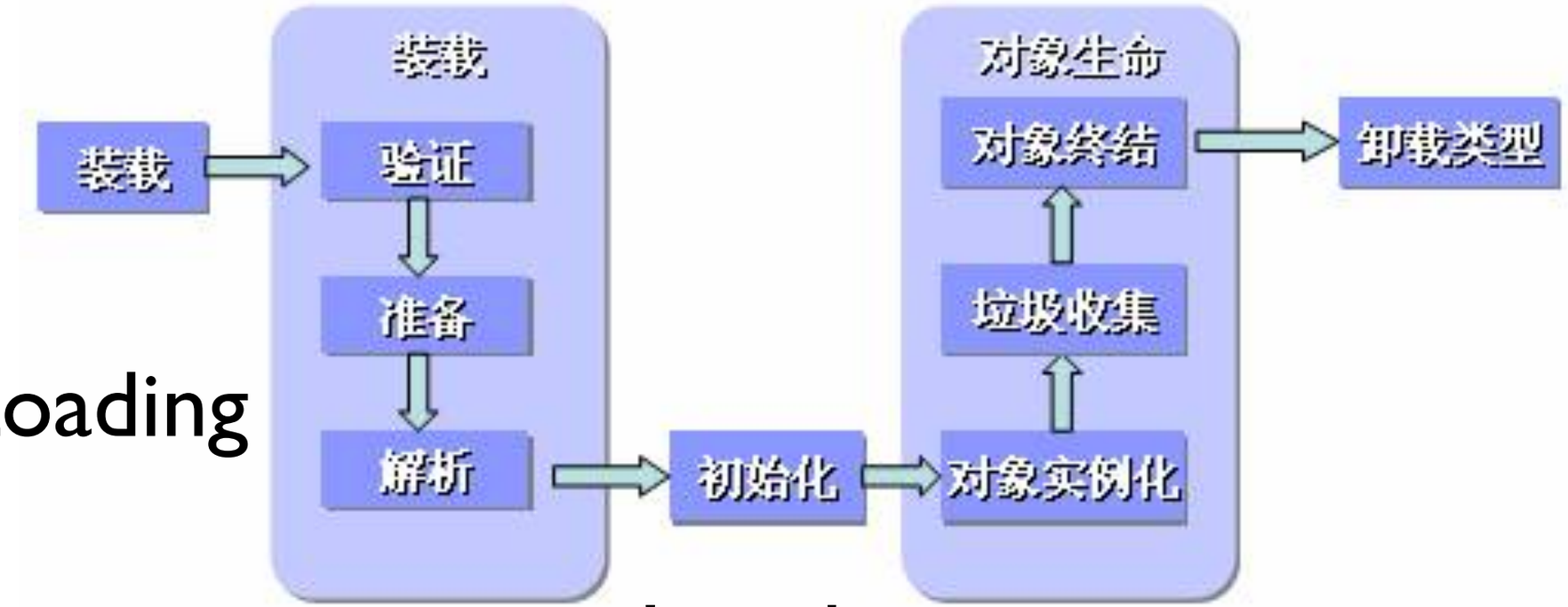
NJUSWI
2002-2012
Anniversary
南京大學 软件学院

简单类的初始化

刘 钦

南京大学软件学院



- Loading
- Initialization
- Linking

类生命周期

静态初始化代码块

```
class StaticDemo{
    static {
        int a=1;
        System.out.println("I am a static block!");
    }
}
```

初始化代码块

- class ConstructorBlockDemo{
- {
- int a=1;
- System.out.println("I am a constructor block!");
- }
- }

构造方法

```
public class ConstructorDemo{
    public ConstructorDemo(){
        int a=1;
        System.out.println("I am a constructor!");
    }
}
```

成员变量

- class FieldDemo{
- 	int b; //initialized implicitly
- 	int a=1; //initialized explicitly
- }

对象初始化初步

1. 静态变量初始化
2. 静态初始化代码块
3. 变量和初始化代码块
4. 构造方法

同一级别，按文字顺序

```
public class InitialOrderTest {
    // 静态变量
    public static String staticField = "静态变量";
    // 变量
    public String field = "变量";
    // 静态初始化块
    static {
        System.out.println(staticField);
        System.out.println("静态初始化块");
    }
    // 初始化块
    {
        System.out.println(field);
        System.out.println("初始化块");
    }
    // 构造器
    public InitialOrderTest() {
        System.out.println("构造器");
    }
    public static void main(String[] args) {
        new InitialOrderTest();
    }
}
```

```
静态变量
静态初始化块
变量
初始化块
构造器
```

静态域的初始化顺序的示例

```
public class StaticOrder{
    public static int X = 20;
    public static int Y = 2 * X;
    static{
        X = 30;
}
 public static void main(String[] args){
    System.out.println(Y);          //输出40 ;
}
```

```
public class StaticTest {
public static int X = 10;
static {

    X = 30;

}

public static int Y = X * 2;

public static void main(String[] args) {

    System.out.println(Y); // 输出60
}
}
```

```
public class StaticTest {
    //这里和上一段代码的不同在于把静态代码块和静态变量X的赋值这两句代码交换了位置。

    static {
        X = 30;
    }
    public static int X = 10;

    public static int Y = X * 2;

    public static void main(String[] args) {

        System.out.println(Y); // 输出20

    }

}
```