# 【基础】

## 变量

1、var ：java SE 10起支持

## Array

1. 初始化：int[] anArray = new int[10];
2. 使用：int[] anArray;
3. 二维：String [][]
4. 复制：arraycopy(...)
5. 内置方法：java.util.Arrays.t

## yield

1、让出当前线程可以让出cpu，如果有其他资源争夺cpu，可以执行其他程序

2、结合switch使用

## class

1. 继承：public class myclass extends me{}

## 接口interface

1. interface myInterface{  
    void method1();  
    void method2();  
   }
2. 使用  
   class my implements myInterface{  
    void method1(){xxx};  
    void method2(){xxx};  
   } // 该类执行接口，必须实现所有接口，否则报错

## package:命名空间

1. 命名空间内的类名不能重复

## record 类

1. 不可变更的数据类（规定的样例代码）

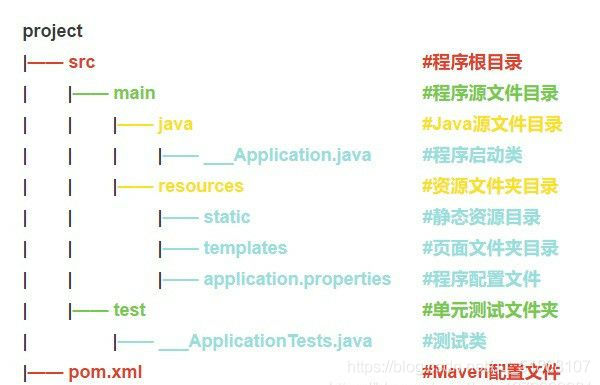
**public** record **User1(**String name**,**String password**)** **{}**

1. 编译时，自动添加get方法，定义final变量，无set 方法，自动重写toString(),equals(),hashCode()方法
2. 只能声明静态属性

# 类路径：

# 【spring Boot】

1. 项目结构



1. 主要目录

| **目录名称** | **相对路径** | **主要用途** |
| --- | --- | --- |
| 源码目录 | src\main\java | 存储源码 |
| 资源目录 | src\main\resources | 存储静态资源、动态页面、配置文件 |
| 测试目录 | src\test\java | 存储单元测试、测试程序 |

1. 启动类：applicaiont.java
   1. @SpringBootApplication此注解标识此类为一个springboot项目启动类
      1. 启动时扫码启动类目录下的所有子类
      2. 可修改
   2. 其他子类扫描：@ComponentScan(basePackages = { “com.xiong.test” })
      1. 主要是用加载当前项目自身开发的bean）
      2. 扫包 + @Service、@Controller、@Component
   3. @SpringBootConfiguration
      1. 配置类 + @Bean
      2. 配置类作用：方便扩展、方便集成，不受包约束。
   4. @EnableAutoConfiguration
      1. springboot内部的starter加载的原理
      2. 内部其实：配置+@Bean
2. 类：Spring boot自动注入的是无参数的构造方法，若需要构造有参数构造方法，需要额外增加1个无参数构造方法
3. target：项目编译后自动生成的项目文件，使用maven打包后的文件也会在此处。
4. Pom.xml：此文件中的重点为dependencies节点配置的各种starter。

## Bean

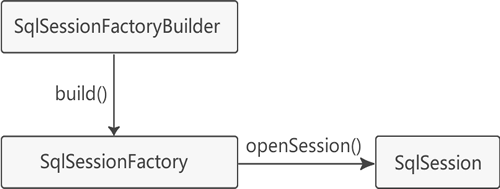
1. 概念：IOC容器的实例化，组装和管理的对象，程序是由1个个的bean组成
   1. Spring IOC 容器：所有对象由IOC创建，低耦合性

## 注解

1. @Required:依赖检查
2. @Autowired：按类型装配。用于替代基于XML配置的自动装配
   1. 默认根据类型注入，可以用于构造器、字段、方法注入
3. @Value：注入int、float、String等基本数据类型，只能标注在成员变量、setter方法上。

# 【 Mybatis】

1. 基础
   1. 基础类



* 1. 主配置文件：mybatis-config.xml //位于resources 路径

configuration配置

properties 可以配置在Java属性配置文件中

settings 修改Mybatis在运行时的行为方式

typeAliases 为Java类型命名一个别名（简称）

typeHandlers 类型处理器

objectFactory 对象工厂

plugins 插件

environments 环境

environment 环境变量

transactionManager 事务管理器

dataSource 数据源

mappers 映射器

示例：

<properties resource="application.yml"></properties> //引入spring 配置文件

<dataSource type="POOLED"> //定义链接数据库

<property name="driver" value="${driver-class-name}"/>

<property name="url" value="${url}"/>

<property name="username" value="${username}"/>

<property name="password" value="${password}"/>

</dataSource>

// ${driver-class-name} 为引用application.yml 文件中的 application.yml 属性

<mappers>

<mapper resource="Mapper/WebsiteMapper.xml"/>

</mappers> //引入XML 映射语句

* 1. XML 映射语句 ：写SQL语句
  2. 从 XML 中构建 SqlSessionFactory：从xml文件中创建