面向对象基础

封装

- 优点:
  - 减少耦合:
  - 可以独立地开发、测试、优化、使用、理解和修改 减轻维护的负担:
  - 可以更容易被程序员理解，并且在调试的时候可以不影响其他模块 有效地调节性能:
  - 可以通过剖析确定哪些模块影响了系统的性能 提高软件的可重用性 降低了构建大型系统的风险:
  - 即使整个系统不可用，但是这些独立的模块却有可能是可用的

```
package basicPackage;

import java.util.Date;

public class Person {
    private String name;
    private Integer age;
    private Date birthday;

    /*省略getter setter*/
}

```

- get方法 获得东西 理解为得到对象的某个属性
- set方法 设置东西 理解为为对象设置某个属性
- tostring 将类设置为String 输出的时候好看、好观察 本质是一个强制转换为String的方法

继承

继承实现了一种 IS-A 的关系

- 例子 给person加一个方法，写一个类继承person

多态

多态是同一个行为具有多个不同表现形式或形态的能力

```
import basicPackage.Person;
import basicPackage.Worker;
import org.junit.Test;

public class BasicTest {

    @Test
    public void testBasic() {
        Person person = new Person();
        Person worker = new Worker();

        person.introduce();
        worker.introduce();
```

```
14
15          person.setAge(20);
16          worker.setAge(16);
17
18          System.out.println(person.getAge());
19          System.out.println(worker.getAge());
20      }
21  }
```

工具详解 - Maven项目构建

- maven项目最重要的是结构
  - 根目录(src):
    - main:
      - java:
      - resources:
      - ...
    - test:
  - pom.xml
- pom.xml

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <project xmlns="http://maven.apache.org/POM/4.0.0"
3           xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4           xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
   http://maven.apache.org/xsd/maven-4.0.0.xsd">
5
6      <modelVersion>4.0.0</modelVersion>
7
8      <groupId>org.example</groupId>
9      <artifactId>project</artifactId>
10     <version>1.0-SNAPSHOT</version>
11
12     <dependencies>
13         <dependency>
14             <groupId>junit</groupId>
15             <artifactId>junit</artifactId>
16             <version>4.13</version>
17             <scope>test</scope>
18         </dependency>
19     </dependencies>
20
21     <properties>
22         <maven.compiler.source>11</maven.compiler.source>
23         <maven.compiler.target>11</maven.compiler.target>
24     </properties>
25
26 </project>
```

- 这是最基本的空的maven的配置文件，可以看到一个约束头，没人记得住，反正都是idea帮忙生成的，没有生成就去复制
- 然后看到第一个 modelVersion 它就是4.0.0 规定死的 现在都用 maven2、maven3 它们的version就是4.0.0

- 然后是groupId那一坨 可以看到 我没改动就是默认长这样的 说明java希望大家用com.xxx.xxx去设置文件的层级、取名 一般我会叫什么（makotogu）github的昵称， 然后是artifact就是你项目的名字，然后version就是版本了
- dependency 这个就是依赖管理了，maven好用的地方来了 众所周知java很多功能都有别人帮忙完成，我们去偷一下就行，从哪里去偷？没有maven、gradle这种工具只能从网上去下载，下载完之后添加到库再去使用，麻烦。maven提供了巨多的依赖，他提供了官方的依赖仓库，https://mvnrepository.com/ 这个地方就是官方的依赖了，复制黏贴一下就可以用了。
- properties 这个怎么说呢，其实有很多用法，但是这里基础就只需要去写个这个，这样可以避免一部分时候idea抽风，用错jdk的版本减少一点报错的可能。
- 还有一个plugins 可以在maven的功能里加什么tomcat启动什么的 快捷一点省力一点，没怎么特别用过，大概是idea太好了吧
- maven会一直存在，它超好的，但不排除如果我学了gradle变心了，那就会再写gradle怎么去用。
- 后面的ssm也都是用maven做管理的

服务器 tomcat

- 这个说难不难，说简单不简单。 汤姆猫?
- 这是一个开源的服务器，好多都用这个，配置完一大坨就可以在电脑上启动web服务，这个电脑就不是普通的电脑了就是一台服务器了。在同一个局域网应该是可以去多设备访问的，如果访问不到也蛮正常的，可能防火墙没设置，可能配置关掉了，具体情况具体分析。
- 下载就不说了 但建议不要用9以后的版本，有些包貌似还不支持，等支持我就改。（java还行，用python最新的的话pip版本支持少得可怜直接死亡）
- 展开叙述生命周期，设计思想比较的复杂，可以考虑先会做事，再去理解，打算放在能做一整个页面之后再去解释

基本语法