# 1 框架

**让程序员专注于业务逻辑, 进而提升开发效率.**

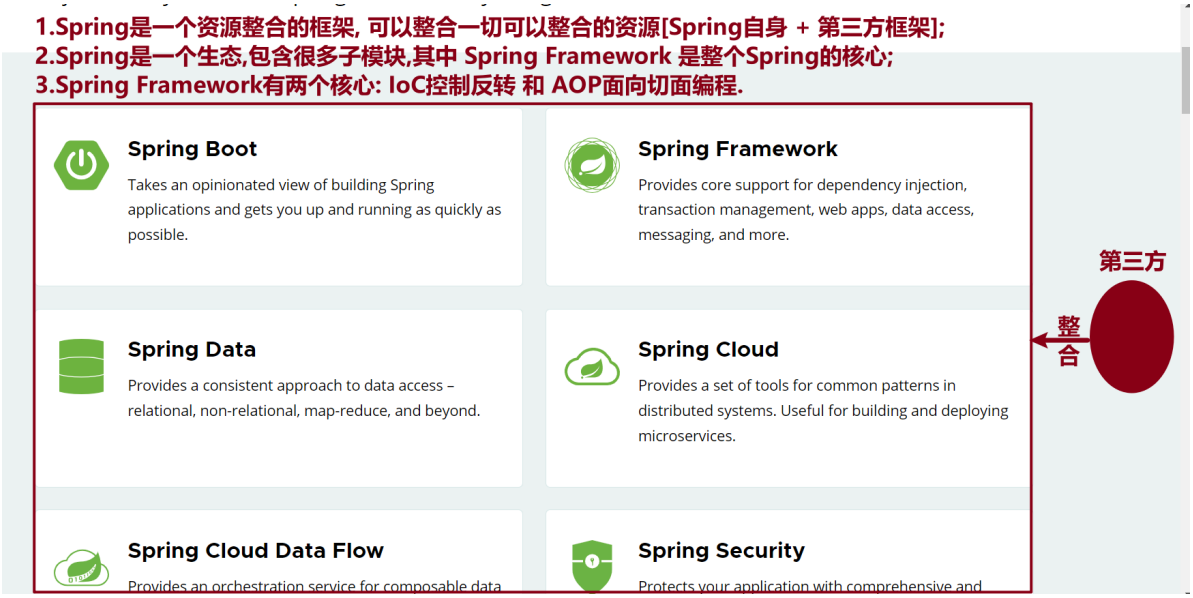
框架的主要作用是帮助开发人员快速、高效地开发应用程序，提供一套完整的系统结构、规范的开发流程、通用的功能和模块、配置文件管理、错误和异常管理以及数据库支持等，为开发人员提供了便利的开发工具和方法。

- Java相关框架: `Spring`
- Python相关框架: `Django、Flask、Tornado`

# 2 Spring框架

官网: https://spring.io

## 2.1 定义




## 2.2 Spring版本

- Spring6: 只支持 `JDK17` 及以上版本
- Spring5: 支持 `JDK8-15` 版本,课程中采用: **5.3.24**

## 2.3 使用流程-注解方式

- 第1步: 添加依赖, 刷新 `maven`;
- 第2步: 在指定类上添加注解: `@Component`
- 第3步: 创建测试类

```
//1.创建IoC容器;
ApplicationContext context = new AnnotationConfigApplicationContext("包路径");
//2.获取Bean;
context.getBean(类型.class);
```

## 3 对象

- `Java` 对象: 开发人员手动创建的对象, 叫做 `Java` 对象.
- `Spring Bean` 对象: 由 `Spring` 框架创建的对象叫做 `Spring Bean` 对象.

  **这两种对象在使用上没有任何差别, 只是为了区分对象创建的方式.**

## 4 相关概念

- IoC

  控制反转的编程思想, 反转资源的获取方向;

  把对象的创建和管理交由框架来完成, 而不是由开发人员手动创建和管理.

- IoC容器

  实现IoC控制反转思想的一种技术手段.

  创建IoC容器

  ```
  ApplicationContext context = new AnnotationConfigApplicationContext("包路径");
  ApplicationContext context = new ClasspathXmlConfigApplicationContext("xxx.xml");
  ```

- 依赖注入-DI

  给 `Spring Bean` 对象的属性赋值.

  **IoC是控制反转思想, IoC容器和DI依赖注入是实现IoC控制反转思想的两种技术手段.**

## 步骤

- 第1步：添加 `Spring Framework` 依赖，刷新 `Maven`;
- 第2步：创建 `Java` 类;
- 第3步：通知框架帮我们创建对象;
- 第4步：获取对象并直接使用.