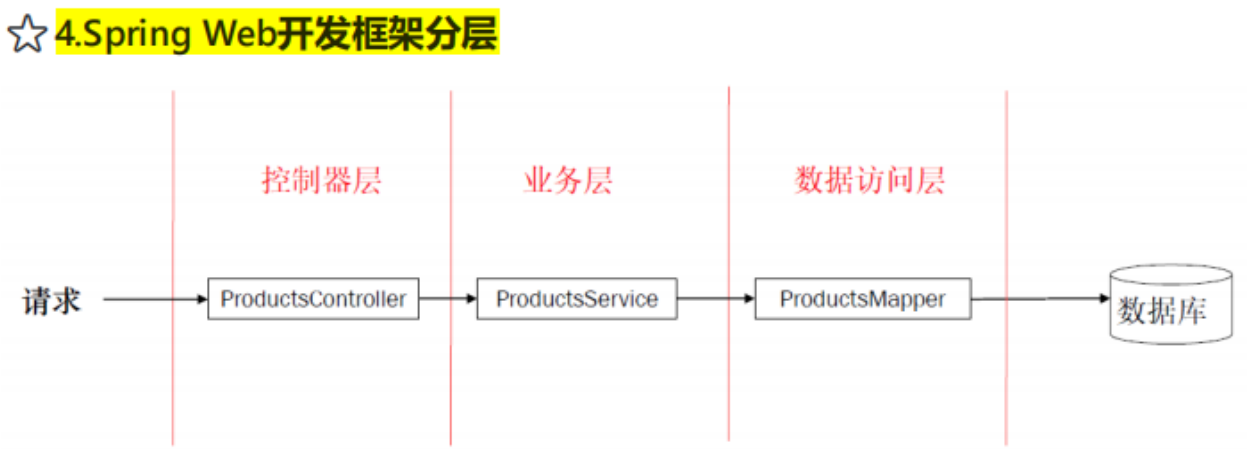
服务端必考

开发环境

源代码仓库管理

- 也称为版本控制（version control）系统，常用工具有：GitLab、SVN（Subversion）、Bitbucket等；
- 🌶️ 需纳入版本控制的有：功能代码、测试代码、测试脚本、构建脚本、部署脚本、配置文件等；
- 🌶️ 从暂存区（index）提交到本地仓库使用的命令是 `git commit`。

Spring Web开发框架分层



自动化测试框架：JUnit

- @TEST
- 没有返回值
- 没有参数

🌶️ 开发期工具：**Spring Boot DevTools**

- 代码变更后应用会自动重启（需要借助 IDE 的自动编译）
- 当面向浏览器的资源（如模板、JavaScript、样式表）等发生变化时，会自动刷新浏览器
  - 应用会暴露 LiveReload 端口，日志如：`LiveReload server is running on port 35729`
  - 需要安装 VSCode 插件 LiveReload （IntelliJ IDEA 要做的配置见下页 ppt）
  - 需要安装浏览器插件：LiveReload，并打开
- 自动禁用（页面渲染的）模板缓存
- 如果使用 H2 数据库，则内置了 H2 控制台。访问：http://localhost:8080/h2-consle

- 🌶️ 强调：该工具只在运行期使用，所以依赖包中它的依赖范围是 Runtime，与编译器无关，不会有编译优化。

依赖注入

A对象注入B对象的方式

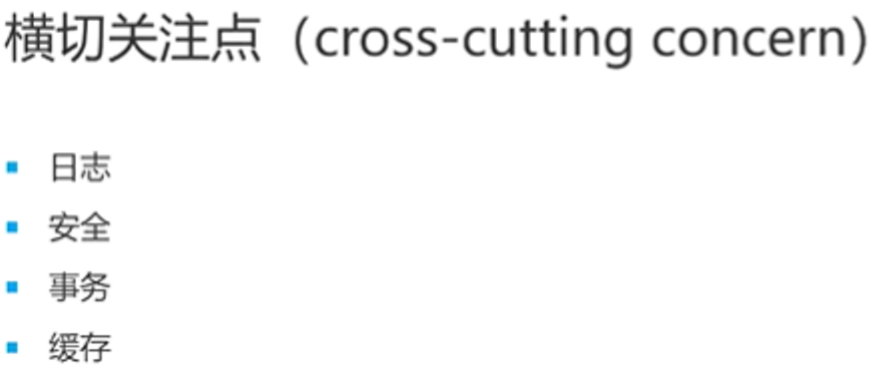
1. 作为构造函数的参数
2. set方法的参数
3. 在公有属性或者私有属性上加@autowired注解

Spring配置

- 自动化配置：涉及到的注解
  - @Component
  - @ComponentScan
    - 会递归的搜子包的路径
  - @Autowired
  - @Configuration
- JavaConfig
  - @Bean
- XML—考试不考

AOP

切面关注点



AOP术语

- 通知（Advice）：切面做什么以及何时做
- 切点（Pointcut）：何处
- 切面（Aspect）：Advice 和 Pointcut 的结合
- 连接点（Join point）：Spring 切面可以在方法前后连接，不可以在字段修改、构造方法上连接
- 引入（introduction）：引入新的行为和状态
- 织入（Weaving）：切面应用到目标对象的过程

AspectJ 切点指示器（pointcut designator）的5种效果

- 指定在哪些方法上切
- 可以通过args获取参数
- 指定包路径
- 指定Bean名称
- 限定指定注解

MVC

MVC

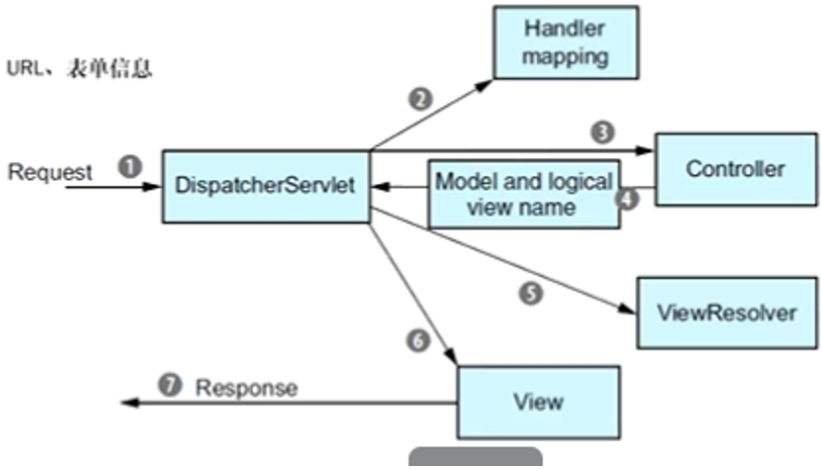
- model-view-controller
- 模型(model): 存储内容，指数据、领域类
- 控制器 (controller) : 处理用户输入
- 视图 (view):显示内容

SpringMVC请求处理过程（必考）

1. 根据 url 找到控制器，控制器解析并获取参数，同时将转向业务层进行处理
2. 涉及到数据的持久化则访问数据访问层，访问完毕后返回业务层，业务层处理完毕将结果数据返回到控制器
3. 控制器处理分两种情况：
    - 前后端分离：返回 Java 对象，同时加上 `@ResponseBody` 注解；客户端请求页面、json 格式数据，分别处理路径是什么？
    - 前后端不分离：控制器将数据赋值给 model 的属性，同时返回视图名。根据视图名做视图解析，找到模板路径，通过第三方页面渲染，将 model 数据渲染到最终页面中，返回 html 格式文件。

    在类的上方加注解 `@RestController`。

Spring MVC的请求处理过程



客户端请求参数分类

1. 路径参数，`@PathVariable`
2. 请求参数（查询参数），`@RequestParam`
3. 表单参数，应用于前后端不分离的传统场景，默认，对应 model 对象，可以使用 `@Valid` 校验
4. `json` 请求体，应用于前后端分离的场景，使用 `@RequestBody` 把 `json` 格式转成 java 对象；`@ResponseBody`，把 java 对象转成 `json` 格式

Spring MVC获取参数的几种方式

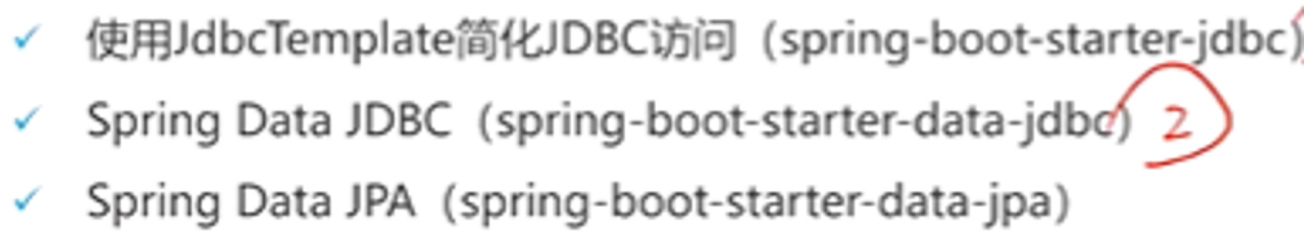
- 表单（form）参数，转成model（成员类型可能会用到Converter进行类型转换），可以使用@Valid校验
- 路径参数，@PathVariable，例子：/book/{id}
- 请求参数（查询参数），@RequestParam，例子：/challenge?mode=2&id=13412431234
- json请求体，@RequestBody，会用到HttpMessageConverter消息转换器，Rest API

JDBC

基于Repository接口的优点：

- 方便进行开发和测试
- 不用在乎实现，可以灵活地替换实现，不用改业务代码

🌶三种关系型数据访问方式的不同（必考）



1. ab需要手动定义scheme脚本来初始化数据，c会根据java对象自动建表
2. a需要手动实现接口，bc不需要手动实现接口（不需要写代码）
3. 数据对象：a不需要加注解，bc需要加注解来将领域对象和数据库对象绑定
4. bc支持@Query注解，但是jpa（C）更加灵活
5. id字段的处理：a：数据库自动生成id，然后手动获取；但是bc不需要操心id
6. b注解是java spring规范@Ta ble；而C里的注解是jpa规范：javax.persistence @Entity
7. 接口不变，改变实现（选择/判断）

Model

- 是页面数据的输入
- Model属性会复制到Servlet Request属性中，这样视图中就可以使用它们用于渲染页面

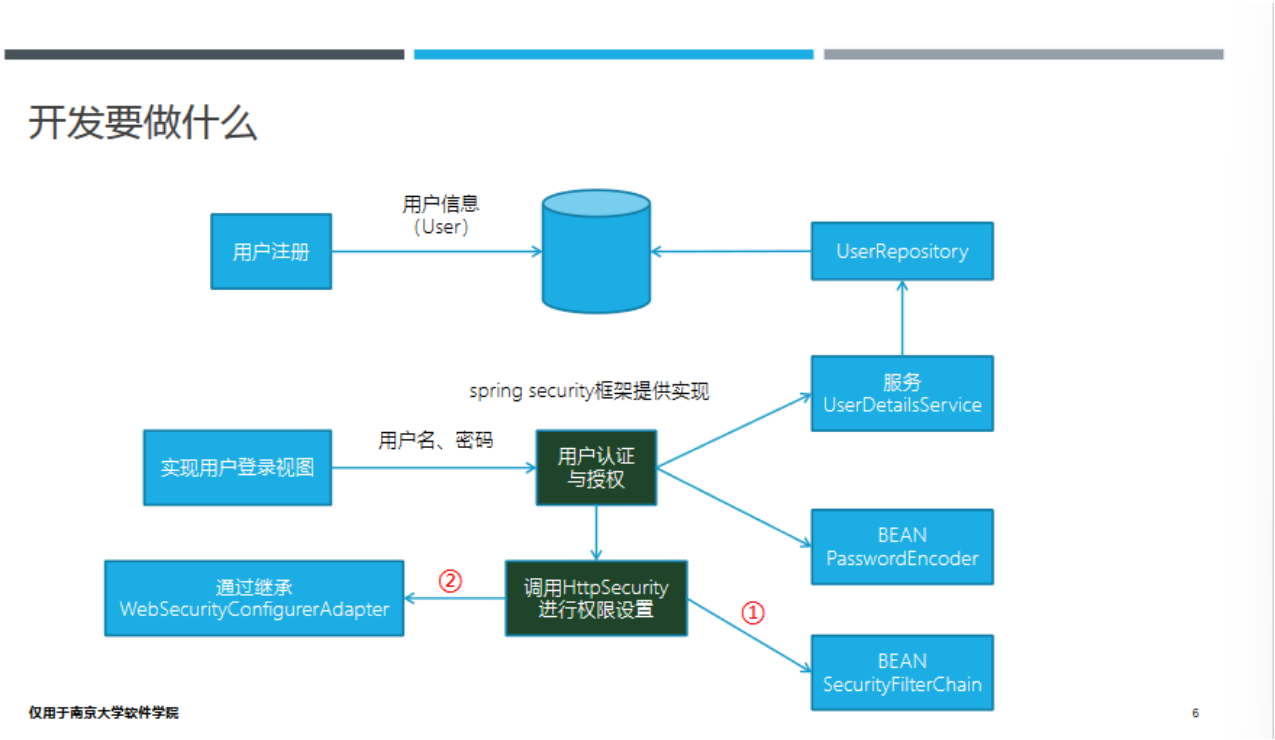
非关系型数据库

🌶Redis（必考）：

- key-value的Hash表结构，value是某数据结构(🌶String,List,Hash,Set)
- 内存数据库 （缓存）
- 集群
  - 冗余存储-提高安全性
  - 加快读写速度
- 主从（master/slave）复制，主：写，从：读
- 数据持久化
- 注意key、value区分大小写
- RedisTempelate<key类型，元素的基本类型（元素or
- 本身）>
  - opForValue，opsForSet
- 序列化：Jackson2JsonSerializer

Security

框架



登录和权限控制

userDetailService接口：给框架提供用户的信息。

PasswordEncorder密码加解密

登陆页面：提供用户认证Controller，

权限设置：通过FilterChain，基于HttpSecurity进行权限设定。Adapter

实现登陆控制器：get，post

请求会重定向到登陆页面

通过filter对权限进行控制

无权限前段会报错，后端是info级别报错

用户信息存储：内存、JDBC、LDAP目录数据库。

权限：关键字= ROLE_+角色名。指这个角色的权限

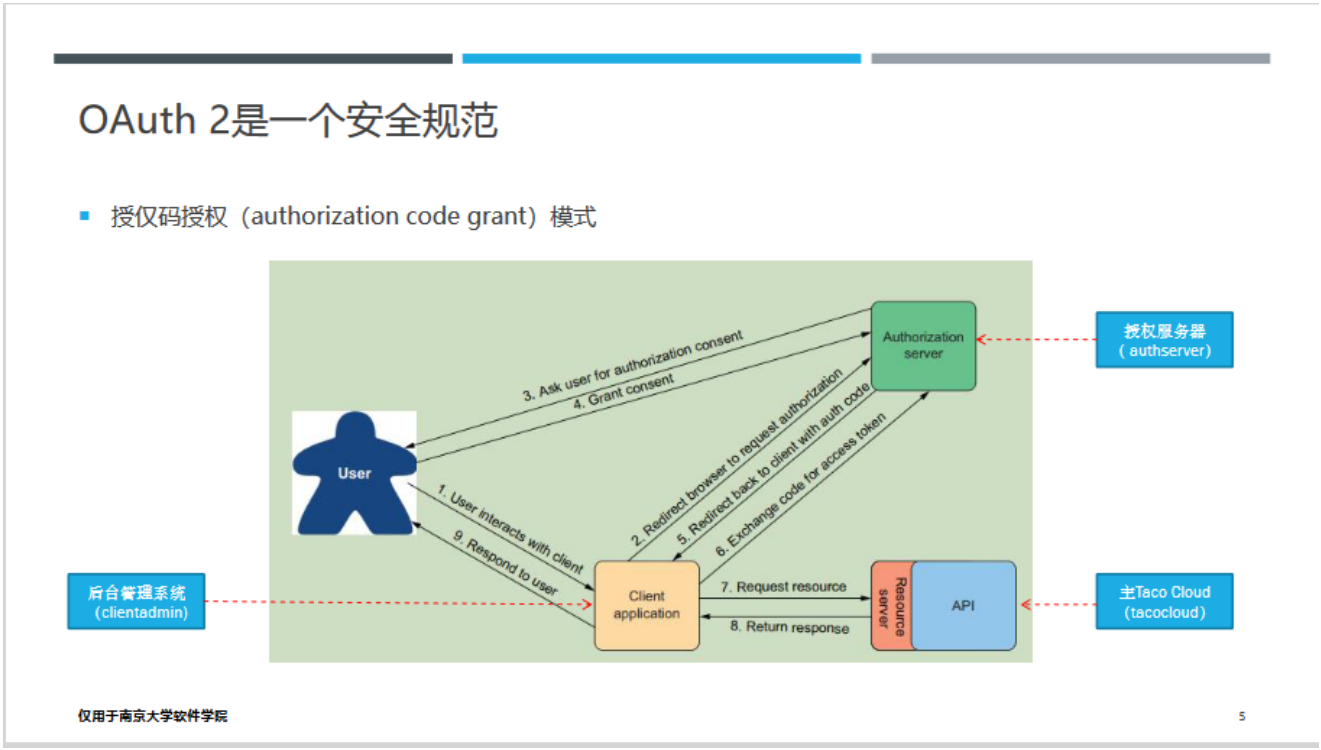
自定义登陆页面：要告诉spring你的页面和Controler和usernakme和password字段

RestfulAPI

接口设计（必考）

OAuth2

授权码授权模式（必考）



\1. 用户访问客户端（第三方应用程序）

\2. 客户端发现用户未登录，将本次访问重定向到授权服务器

\3. 授权服务器向用户索取用户名密码

\4. 用户向授权服务器提供用户名密码后通过验证

\5. 授权服务器给客户端一个 Auth Code

\6. 客户端通过Auth Code向授权服务器请求一个 Acess Token

\7. 客户端携带Acess Token向资源服务器访问资源

\8. 资源服务器向客户端返回资源

\9. 客户端将资源展示给用户

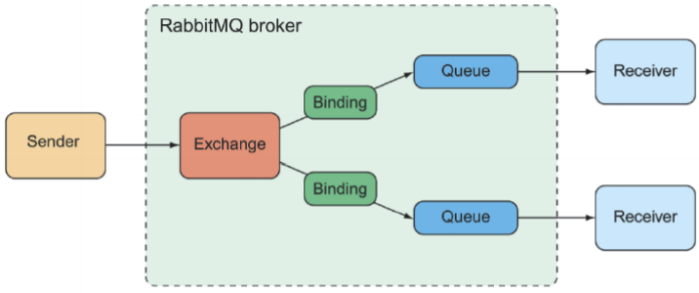
ActiveMQ、RabbitMQ

RabbitMQ（必考）

4. RabbitMQ

- 实现AMQP(Advanced Message Queueing Protocol)协议

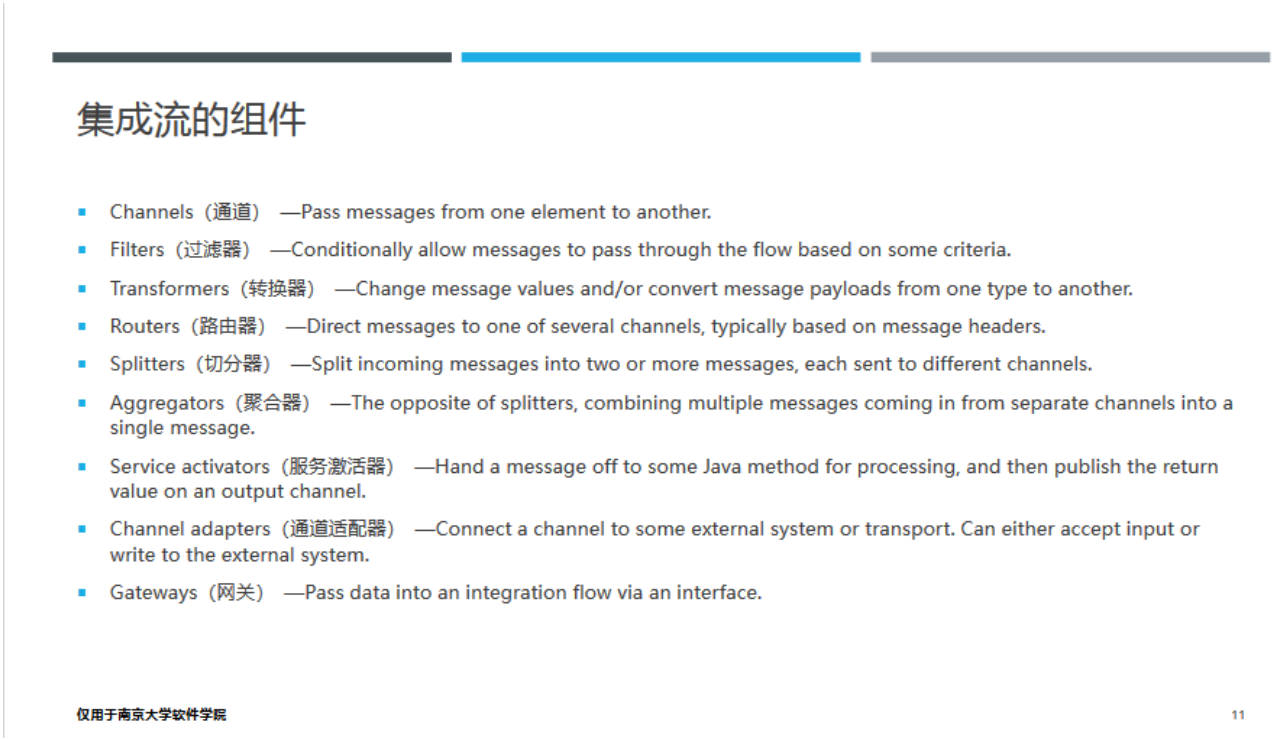
4.1 概念:



- Sender：生产方 Receiver：消费方
- Queue：消息队列
- Exchange：“交换机”，交换机决定消息发往哪个队列（消息路由）
  - Routing key：字符串，来自于Sender，告诉Exchange该消息发往哪个队列
  - Binding key：字符串，创建队列时定义该队列的key
  - 规则：Default、Direct（直接转发）、Topic、Fanout（广播）
  - Headers、Dead letter（不重要）

集成流

组件（必考）



通道Channel：传输信息

过滤器Filter：使消息按某种条件通过

转换器Transformer：改变消息值、将消息载体的类型更改

路由器Router：多通道时定义消息该通向哪里

切分器Splitter：将一个消息切分成两个或多个消息发往不同通道

聚合器Aggregator：与切分器相反。多通道消息组合成一个消息

服务激活器Service Activator：触发一个Java方法并返回到Channel

通道适配器Channel Adapter：将Channel连接至某个系统或中转，入或出都需要

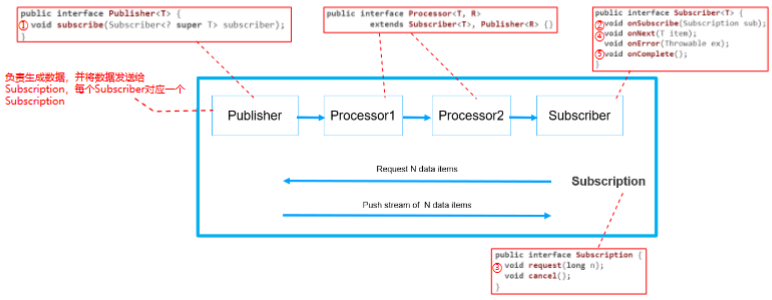
网关Gateway：将数据传入集成流的接口

反应式编程

反应式流规范定义的4个接口（必考）

反应式流规范定义的4个接口

- org.reactivestreams.*



两个基本概念：Flux 和 Mono

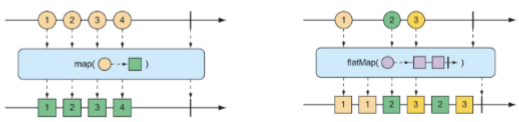
- Flux：包含 0 到 N 个元素的异步序列
- Mono：包含 0 或者 1 个元素的异步序列
- 消息：正常的包含元素的消息、序列结束的消息和序列出错的消息
- 操作符（Operator）：对流上元素的操作

转换flatMap：返回Momo类型（必考）

转换Flux流1

- map，同步
- flatMap，异步

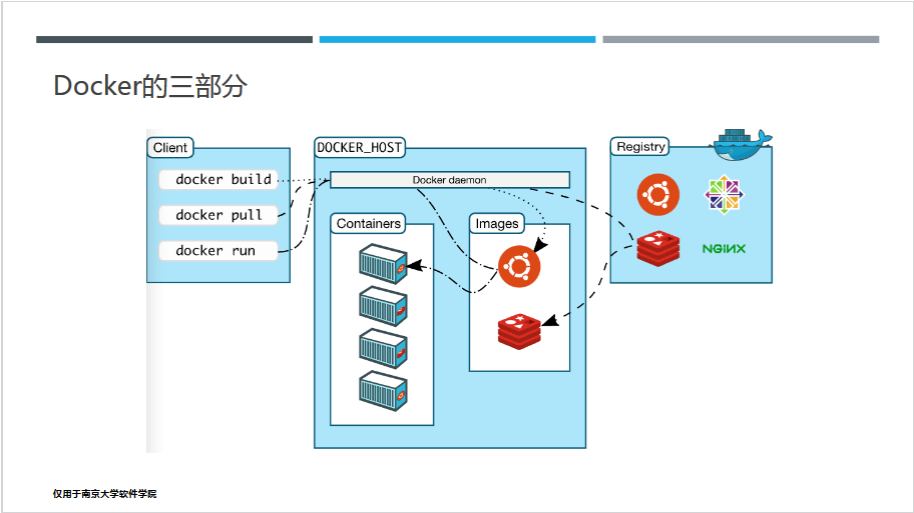
- 并发模型（Schedulers方法）
  - ✓ .immediate()
  - ✓ .single()
  - ✓ .newSingle()
  - ✓ .elastic()
  - ✓ .parallel()



Reactor项目

Docker

docker三个部分（必考）



docker-compose ps和images：都是用来检测当前目录下的运行的服务的容器和镜像

容器镜像构建与编排

由Dockerfile构建镜像