什么是 ssh?

SSH 对应 struts spring hibernate

struts 采用 MVC 模式，主要是作用于用户交互

spring 采用 IOC 和 AOP~作用比较抽象，是用于项目的松耦合

hibernate 是对象持久化框架，其实就是实体类和数据库表建立关系，操作类就会触发相应的 sql 语句，可以不用写任何 sql 语句，完成数据库编程

SSH 就是 Struts + Spring + Hibernate 3 个 Java 框架的集合，现在 Java 开发中常用的框架组合。用来开发后台，与前台和数据库进行数据交互。

Hibernate 工作原理及为什么要用？

原理：hibernate，通过对 jdbc 进行封装，对 java 类和 关系数据库进行 mapping，实现了对关系数据库的面向对象方式的操作，改变了传统的 jdbc + sql 操作数据的方式，从而使开发人员可以话更多精力进行对象方面的开发。

为什么要用：1.对 JDBC 访问数据库的代码做了封装，大大简化了数据访问层繁琐的重复性代码。2.Hibernate 是一个基于 JDBC 的主流持久化框架，是一个优秀的 ORM 实现。他很大程度的简化 DAO 层的编码工作 3.hibernate 的性能非常好，因为它是个轻量级框架。映射的灵活性很出色。它支持各种关系数据库，从一对一到多对多的各种复杂关系。

Hibernate 是如何延迟加载?

Hibernate3 提供了属性的延迟加载功能。当 Hibernate 在查询数据的时候，数据并没有存在于内存之中，而是当程序真正对数据的操作时，对象才存在于内存中，就实现了延迟加载，他节省了服务器的内存开销，从而提高了服务器的性能。

Hibernate 的对象有几种状态

瞬时态（transient） 持久态（persistent） 游离态（detached）

Struts 工作机制？为什么要使用 Struts?

Struts 的工作流程:

在 web 应用启动时就会加载初始化 ActionServlet,ActionServlet 从

struts-config.xml 文件中读取配置信息,把它们存放到各种配置对象

当 ActionServlet 接收到一个客户请求时,将执行如下流程.

-(1)检索和用户请求匹配的 ActionMapping 实例,如果不存在就返回请求路径无效信息;

-(2)如果 ActionForm 实例不存在,就创建一个 ActionForm 对象,把客户提交的表单数据保存到 ActionForm 对象中;

-(3)根据配置信息决定是否需要表单验证.如果需要验证,就调用 ActionForm 的 validate()方法;

-(4)如果 ActionForm 的 validate()方法返回 null 或返回一个不包含 ActionMessage 的 ActuibErrors 对象, 就表示表单验证成功;

-(5)ActionServlet 根据 ActionMapping 所包含的映射信息决定将请求转发给哪个 Action,如果相应的 Action 实例不存在,就先创建这个实例,然后调用 Action 的 execute()方法;

-(6)Action 的 execute()方法返回一个 ActionForward 对象,ActionServlet 在把客户请求转发给 ActionForward 对象指向的 JSP 组件;

-(7)ActionForward 对象指向 JSP 组件生成动态网页,返回给客户;

为什么要用:

1. JSP、Servlet、JavaBean 技术的出现给我们构建强大的企业应用系统提供了可能。但用这些技术构建的系统非常的繁乱。

2. 基于 Struts 开发的应用:不用再考虑公共问题 专心在业务实现上 结构统一，易于学习、维护

为什么要用 spring?

Spring 是一个轻量级的 IOC 和 AOP 框架。IOC（控制反转）意味着将你设计好的类交给系统去控制，而不是在你的类内部控制。这称为控制反转。AOP（面向切面），它将那些影响多个类的行为封装到可重用的模块中，面向对象是把问题从同类事物中抽象出来，面向切面是把问题从不同类问题中抽象出来。

Struts 的优缺点

优点：

1. 实现 MVC 模式，结构清晰,使开发者只关注业务逻辑的实现.

2. 有丰富的 tag 可以用 ,Struts 的标记库(Taglib)，如能灵活动用，则能大大提高开发效率。另外，就目前国内的 JSP 开发者而言，除了使用 JSP 自带的常用标记外，很少开发自己的标记，或许 Struts 是一个很好的起点。

3. 页面导航.页面导航将是今后的一个发展方向，事实上，这样做，使系统的脉络更加清晰。通过一个配置文件，即可把握整个系统各部分之间的联系，这对于后期的维护有着莫大的好处。尤其是当另一批开发者接手这个项目时，这种优势体现得更加明显。

4. 提供 Exception 处理机制 .

5. 数据库链接池管理

6. 支持 I18N

缺点：

1.转到展示层时，需要配置 forward，每一次转到展示层，相信大多数都是直接转到 jsp，而涉及到转向，需要配置 forward，如果有十个展示层的 jsp，需要配置十次 struts，而且还不包括有时候目录、文件变更，需要重新修改 forward，注意，每次修改配置之后，要求重新部署整个项目，而 tomcate 这样的服务器，还必须重新启动服务器，如果业务变更复杂频繁的系统，这样的操作简单不可想象。现在就是这样，几十上百个人同时在线使用我们的系统，大家可以想象一下，我的烦恼有多大。

2. Struts 的 Action 必需是 thread－safe 方式，它仅仅允许一个实例去处理所有的请求。所以 action 用到的所有的资源都必需统一同步，这个就引起了线程安全的问题。

3. 测试不方便. Struts 的每个 Action 都同 Web 层耦合在一起，这样它的测试依赖于 Web 容器，单元测试也很难实现。不过有一个 Junit 的扩展工具 Struts TestCase 可以实现它的单元测试。

4.类型的转换.Struts 的 FormBean 把所有的数据都作为 String 类型，它可以使用工具 Commons-Beanutils 进行类型转化。但它的转化都是在 Class 级别，而且转化的类型是不可配置的。类型转化时的错误信息返回给用户也是非常困难的。

5. 对 Servlet 的依赖性过强. Struts 处理 Action 时必需要依赖 ServletRequest 和 ServletResponse,所有它摆脱不了 Servlet 容器。

6. 前端表达式语言方面.Struts 集成了 JSTL，所以它主要使用 JSTL 的表达式语言来获取数据。可是 JSTL 的表达式语言在 Collection 和索引属性方面处理显得很弱。

7. 对 Action 执行的控制困难. Struts 创建一个 Action，如果想控制它的执行顺序将会非常困难。甚至你要重新去写 Servlet 来实现你的这个功能需求。

8. 对 Action 执行前和后的处理. Struts 处理 Action 的时候是基于 class 的 hierarchies，很难在 action 处理前和后进行操作。
9. 对事件支持不够.
在 struts 中，实际是一个表单 Form 对应一个 Action 类(或 DispatchAction)，换一句话说：在 Struts 中实际是一个表单只能对应一个事件，struts 这种事件方式称为 applicationevent，application event 和 component

什么是 AOP?

面向切面编程（AOP）完善 spring 的依赖注入（DI），
面向切面编程在 spring 中主要表现为两个方面 ：
1.面向切面编程提供声明式事务管理
2.spring 支持用户自定义的切面

什么是 IOC?

不创建对象，但是描述创建它们的方式。在代码中不直接与对象和服务连接，但在配置文件中描述哪一个组件需要哪一项服务。容器（在 Spring 框架中是 IOC 容器） 负责将这些联系在一起。
就是由容器控制程序之间的关系，而非传统实现中，由程序代码直接操控，控制权由应用代码中转到了外部容器，控制权的转移，就是所谓的反转。

Spring IOC （控制反转，依赖注入）

Spring 支持三种依赖注入方式，分别是属性（Setter 方法）注入，构造注入和接口注入。Spring 支持 setter 注入和构造器注入，通常使用构造器注入来注入必须的依赖关系，对于可选的依赖关系，则 setter 注入是更好的选择，setter 注入需要类提供无参构造器或者无参的静态工厂方法来创建对象。
在 Spring 中，那些组成应用的主体及由 Spring IOC 容器所管理的对象被称之为 Bean。
Spring 的 IOC 容器通过反射的机制实例化 Bean 并建立 Bean 之间的依赖关系。
简单地讲，Bean 就是由 Spring IOC 容器初始化、装配及被管理的对象。
获取 Bean 对象的过程，首先通过 Resource 加载配置文件并启动 IOC 容器，然后通过 getBean 方法获取 bean 对象，就可以调用他的方法。

Spring Bean 的作用域：

Singleton：Spring IOC 容器中只有一个共享的 Bean 实例，一般都是 Singleton 作用域。
Prototype：每一个请求，会产生一个新的 Bean 实例。
Request：每一次 http 请求会产生一个新的 Bean 实例。

什么是 DWR?它有哪些功能?

DWR(Direct Web Remoting)是一个 WEB 远程调用框架.
可以在客户端利用 JavaScript 直接调用服务端的 Java 方法并返回值给 JavaScript
DWR 根据 Java 类来动态生成 JavaScrip 代码.
支持 Dom Trees,支持 Spring,支持 commons-logging

BeanFactory 的作用是什么？

BeanFactory 是配置、创建、管理 bean 的容器，有时候也称为 bean 上下文。Bean 与 bean 的依赖关系，也是由 BeanFactory 负责维护的。

什么是过滤器？怎么创建一个过滤器

过滤器：在请求发送之后，处理之前对请求的一次拦截，可以更改请求状态或者参数值等。
创建过滤器：实现 filter 接口，重写 doFilter 方法，最后在 web.xml 中配置过滤器

什么是 Ajax?

AJAX（Asynchronous JavaScript and XML），它不是一门新的语言或技术，而是多种技术的综合，包括：
Javascript XHTML CSS DOM XML XSTL XMLHttpRequest
通过在后台与服务器进行少量数据交换，AJAX 可以使网页实现异步更新。这意味着可以在不重新加载整个网页的情况下，对网页的某部分进行更新。传统的网页（不使用 AJAX）如果需要更新内容，必需重载整个网页面。

同步与异步的区别？

普通 B/S 模式（同步）AJAX 技术（异步）
同步：提交请求->等待服务器处理->处理完毕返回 这个期间客户端浏览器不能干任何事，而异步则是 请求通过事件触发->服务器处理->处理完毕
同步是阻塞模式，异步是非阻塞模式。
同步(发送方发出数据后，等接收方发回) 异步(发送方发出数据后，不等接收方发回响应)

GET 还是 POST?

与 POST 相比，GET 更简单也更快，并且在大部分情况下都能用。
然而，在以下情况中，请使用 POST 请求：
无法使用缓存文件（更新服务器上的文件或数据库）
向服务器发送大量数据（POST 没有数据量限制）
发送包含未知字符的用户输入时，POST 比 GET 更稳定也更可靠

何为容器

容器就是符合某种规范的能够提供一系列服务的管理器.

Spring 实现了那几种模式

工厂模式 和 单例模式

Hibernate 实体之间的关联关系的三种形式 一对一关联 一对多关联 多对多关联

Hibernate 中 Session 对象的 load()方法和 get()方法的区别

①记录不存在时 get()方法会返回空（null），而 load()方法将会抛出一个 HibernateException 异常
②load()方法查询数据时会先找 Hibernate 的内部缓存和二级缓存中的现有数据，get()方法在内部缓存中没有打到相对应的数据时装直接执行 SQL 语句进行查询

spring 框架的 7 个模块是什么？

spring AOP --面象切面编程
Spring DAO --数据访问对象
Spring ORM --对象关系影射
Spring Contect -- 上下文配置，向 Spring 框架提供上下文信息
Spring WEB -WEB 上下文模块

Spring WEB-MVC --实现了 MVC
Spring CORE - 核心容器提供 Spring 框架基本功能
简述什么是 ORM
ORM 的全称是 Object-Relational Mapping 翻译成中文就是“对象－关系映射”
ORM 组件的主要功能就是实现实体域对象的持久化并封装数据库访问的细节
ORM 本身并不是一个组件，它是具用某种功能的组件的总称，也可以说是一种框架结构
在通常情况下软件系统由表示层，业务层，持久层和数据库层组成，Struts 属于哪一层？
Struts 属于表示层组件，它的作用主要体现在以下几个方面：
1)输出用户界面和接收用户的输入，实现与用户的交互。
2)调用业务方法，完成业务处理，还要包括处理后的显示工作。
简述什么是 Struts 什么是 spring
Struts 只是一个 MVC 框架（Framework）,用于快速开发 Java Web 应用。Struts 实现的重点在 C(Controller)，包括 ActionServlet/RequestProcessor 和我们定制的 Action,也为 V(View)提供了一系列定制标签（Custom Tag）。但 Struts 几乎没有涉及 M(Model),所以 Struts 可以采用 JAVA 实现的任何形式的商业逻辑。
Spring 是一个轻型容器(light-weight container)，其核心是 Bean 工厂(Bean Factory)，用以构造我们所需要的 M(Model)。在此基础之上，Spring 提供了 AOP（Aspect-Oriented Programming, 面向层面的编程）的实现，用它来提供非管理环境下申明方式的事务、安全等服务；对 Bean 工厂的扩展 ApplicationContext 更加方便我们实现 J2EE 的应用；DAO/ORM 的实现方便我们进行数据库的开发；Web MVC 和 Spring Web 提供了 Java Web 应用的框架或与其他流行的 Web 框架进行集成。
就是说可将两者一起使用，达到将两者自身的特点进行互补。
Struts 有哪些主要功能
1.包含一个 controller servlet，能将用户的请求发送到相应的 Action 对象。
2. JSP 自由 tag 库，并且在 controller servlet 中提供关联支持，帮助开发员创建交互式表单应用。
3. 提供了一系列实用对象：XML 处理、通过 Java reflection APIs 自动处理 JavaBeans 属性、国际化的提示和消息。

Struts 项目的目标是为创建 Java web 应用提供一个开放源代码的 framework。Struts framework 的内核是基于例如 Java Servlets, JavaBeans, ResourceBundles, 和 XML，以及各种 Jakarta Commons 包的标准技术的灵活的控制层。
Struts 提供了它自身的控制器组件，并整合了其他技术，以提供模型和视图。对于模型，同大多数的第三方软件包一样，如 Hibernate, iBATIS, 或者 Object Relational Bridge，Struts 能够和标准数据连接技术相结合，如 JDBC 和 EJB。对于视图，Struts 与 JavaServer Pages 协同工作，包含 JSTL 和 JSF。
接口和抽象类有什么区别?
接口是公开的，不能包含私有的方法或变量，而抽象类是可以有私有方法或私有变量的，实现接口的一定要实现接口里定义的所有方法，而实现抽象类可以有选择地重写需要用到的方法，接口可以实现多重继承，而一个类只能继承一个超类，但可以通过继承多个接口实现多重继承，接口还有标识（里面没有任何方法，如 Remote 接口）和数据共享（里面的变量全是常量）的作用.一般的应用里，最顶级的是接口，然后是抽象类实现接口，最后才到具体类实现。
什么是抽象类
抽象类仅提供一个类型的部分实现。抽象类可以有实例变量，以及一个或多个构造函数。抽象类可以同时有抽象方法和具体方法。一个抽象类不会有实例，这些构造函数不能被客户端调用来创建实例。一个抽象类的构造函数可以被其子类调用，从而使一个抽象类的所有子类都可以有一些共同的实现，而不同的子类可以在此基础上有其自己的实现。
抽象类的用途
具体类不是用来继承的。Scott Meyers 曾指出，只要有可能，不要从具体类继承。
假设有 2 个具体类，类 A 和类 B，类 B 是类 A 的子类，那么一个最简单的修改方案是应当建立一个抽象类（或 java 接口）C，然后让类 A 和类 B 成为抽象类 C 的子类。
抽象类应当拥有尽可能多的共同代码。以提高代码的复用率。
抽象类应当拥有尽可能少的数据。
web 应用程序体系结构是怎样的?
一般分为表示层、业务层、数据存取层
静态类有什么好处?
一个是不用实例化，一个是可以构建静态工厂方法，还有一个是静态内部类实现具体功能 外部类代理 解耦（
最重要的就是降低包的复杂度?
静态方法有什么好处?
（1）在 Java 里，可以定义一个不需要创建对象的方法，这种方法就是静态方法。要实现这样的效果，只需要在类中定义的方法前加上 static 关键字。例如：public static int maximum(int n1,int n2)
使用类的静态方法时，注意：
a）在静态方法里只能直接调用同类中其他的静态成员（包括变量和方法），而不能直接访问类中的非静态成员。这是因为，对于非静态的方法和变量，需要先创建类的实例对象后才可使用，而静态方法在使用前不用创建任何对象。
b）静态方法不能以任何方式引用 this 和 super 关键字，因为静态方法在使用前不用创建任何实例对象，当静态方法调用时，this 所引用的对象根本没有产生。
（2）静态变量是属于整个类的变量而不是属于某个对象的。注意不能把任何方法体内的变量声明为静态，例如：

```
fun()
{
      static  int  i=0;//非法。
}
```

（3）一个类可以使用不包含在任何方法体中的静态代码块，当类被载入时，静态代码块被执行，且之被执行一次，静态块常用来执行类属性的初始化。例如：

```
static
{
}
```

面向对象的特征有哪些方面
1.抽象：

抽象就是忽略一个主题中与当前目标无关的那些方面，以便更充分地注意与当前目标有关的方面。抽象并不打算了解全部问题，而只是选择其中的一部分，暂时不用部分细节。抽象包括两个方面，一是过程抽象，二是数据抽象。
2.继承：
继承是一种联结类的层次模型，并且允许和鼓励类的重用，它提供了一种明确表述共性的方法。对象的一个新类可以从现有的类中派生，这个过程称为类继承。新类继承了原始类的特性，新类称为原始类的派生类（子类），而原始类称为新类的基类（父类）。派生类可以从它的基类那里继承方法和实例变量，并且类可以修改或增加新的方法使之更适合特殊的需要。
3.封装：
封装是把过程和数据包围起来，对数据的访问只能通过已定义的界面。面向对象计算始于这个基本概念，即现实世界可以被描绘成一系列完全自治、封装的对象，这些对象通过一个受保护的接口访问其他对象。
4.多态性：
多态性是指允许不同类的对象对同一消息作出响应。多态性包括参数化多态性和包含多态性。多态性语言具有灵活、抽象、行为共享、代码共享的优势，很好的解决了应用程序函数同名问题。
如何实现对象克隆？
有两种方式：
1.实现 Cloneable 接口并重写 Object 类中的 clone()方法；
2.实现 Serializable 接口，通过对象的序列化和反序列化实现克隆，可以实现真正的深度克隆，
简介 cookie 的有关知识
浏览器与 WEB 服务器之间是使用 HTTP 协议进行通信的，当某个用户发出页面请求时，WEB 服务器只是简单的进行响应，然后就关闭与该用户的连接。因此当一个请求发送到 WEB 服务器时，无论其是否是第一次来访，服务器都会把它当作第一次来对待，这样的不好之处可想而知。为了弥补这个缺陷，Netscape 开发出了 cookie 这个有效的工具来保存某个用户的识别信息，因此人们昵称为“小甜饼”。cookies 是一种 WEB 服务器通过浏览器在访问者的硬盘上存储信息的手段：Netscape Navigator 使用一个名为 cookies.txt 本地文件保存从所有站点接收的 Cookie 信息；而 IE 浏览器把 Cookie 信息保存在类似于 C: \windows\cookies 的目录下。当用户再次访问某个站点时，服务端将要求浏览器查找并返回先前发送的 Cookie 信息，来识别这个用户。
cookies 给网站和用户带来的好处非常多：
1、Cookie 能使站点跟踪特定访问者的访问次数、最后访问时间和访问者进入站点的路径
2、Cookie 能告诉在线广告商广告被点击的次数 ，从而可以更精确的投放广告
3、Cookie 有效期限未到时，Cookie 能使用户在不键入密码和用户名的情况下进入曾经浏览过的一些站点
4、Cookie 能帮助站点统计用户个人资料以实现各种各样的个性化服务
描述 Cookie 和 Session 的作用，区别和各自的应用范围
Cookie 和 Session 都可以用来在多个页面之间共享数据,区别是 Cookie 保存在客户端,可以设置比较长的保存时间.而 Session 保存在服务器端,通常生存时间较短。如果客户端禁用了 Cookie,Cookie 将无法工作，而 session 不受这一影响。Session 保存在服务端，cookie 保存在客户端
Session 保存是对象，cookie 只能保存字符串
Session 不能设置路径，cookie 可以设置保存路径。同一个网站不同网页的 cookie 可以保存到不通的路机构下，彼此是无法相互访问的。 Session 在服务器关闭后会自动消失，cookie 则不会。
Session 与 Cookie 差异：Cookie 可以让服务端跟踪每个客户端的访问，但是每次客户端的访问都必须传回这些 Cookie，如果 Cookie 很多，则无形的增加了客户端与服务端的数据传输量，
而 Session 则很好地解决了这个问题，同一个客户端每次和服务端交互时，将数据存储通过 Session 到服务端，不需要每次都传回所有的 Cookie 值，而是传回一个 ID，每个客户端第一次访问服务器生成的唯一的 ID，客户端只要传回这个 ID 就行了，这个 ID 通常为 NAME 为 JSESSIONID 的一个 Cookie。这样服务端就可以通过这个 ID，来将存储到服务端的 KV 值取出了。
Session 和 Cookie 的超时问题，Cookie 的安全问题：一般来说保密性高、保存时间短的信息适合用 session 来存放，而 Cookie 适合存放需要长期保存的非敏感数据。
会话跟踪技术 ：Session Cookie 表单隐藏域 url 重写
什么是 web 容器？
给处于其中的应用程序组件（JSP，SERVLET）提供一个环境，使 JSP,SERVLET 直接更容器中的环境变量接口交互，不必关注其它系统问题。主要有 WEB 服务器来实现。例如：TOMCAT,WEBLOGIC,WEBSPHERE 等。该容器提供的接口严格遵守 J2EE 规范中的 WEB APPLICATION 标准。我们把遵守以上标准的 WEB 服务器就叫做 J2EE 中的 WEB 容器。
一次完整的 HTTP 事务是怎样的一个过程？
基本流程：
a. 域名解析
b. 发起 TCP 的 3 次握手
c. 建立 TCP 连接后发起 http 请求
d. 服务器端响应 http 请求，浏览器得到 html 代码
e. 浏览器解析 html 代码，并请求 html 代码中的资源
f. 浏览器对页面进行渲染呈现给用户
什么是 JavaScript？
JavaScript 是客户端和服务器端脚本语言，可以插入到 HTML 页面中，并且是目前较热门的 Web 开发语言。同时，JavaScript 也是面向对象编程语言。
在 React 当中 Element 和 Component 有何区别?
简单地说，一个 React element 描述了你想在屏幕上看到什么。换个说法就是，一个 React element 是一些 UI 的对象表示。一个 React Component 是一个函数或一个类，它可以接受输入并返回一个 React element t（通常是通过 JSX ，它被转化成一个 createElement 调用）。
Controlled Component 与 Uncontrolled Component 之间的区别是什么？
React 的核心组成之一就是能够维持内部状态的自治组件，不过当我们引入原生的 HTML 表单元素时（input,select,textarea 等），我们是否应该将所有的数据托管到 React 组件中还是将其仍然保留在 DOM 元素中呢？这个问题的答案就是受控组件与非受控组件的定义分割。受控组件（Controlled Component）代指那些交由 React 控制并且所有的表单数据统一存放的组件。譬如下面这段代码中 username 变量值并没有存放到 DOM 元素中，而是存放在组件状态数据中。任何时候我们需要改变 username 变量值时，我们应当调用 setState 函数进行修改。
而非受控组件（Uncontrolled Component）则是由 DOM 存放表单数据，并非存放在 React 组件中。我们可以使用 refs 来操控 DOM 元素

在什么情况下你会优先选择使用 Class Component 而不是 Functional Component?

在组件需要包含内部状态或者使用到生命周期函数的时候使用 Class Component ，否则使用函数式组件。

如何告诉 React 它应该编译生产环境版本?

通常情况下我们会使用 Webpack 的 DefinePlugin 方法来将 NODE_ENV 变量值设置为 production。编译版本中 React 会忽略 propType 验证以及其他的告警信息，同时还会降低代码库的大小，React 使用了 Uglify 插件来移除生产环境下不必要的注释等信息。

概述下 React 中的事件处理逻辑

为了解决跨浏览器兼容性问题，React 会将浏览器原生事件（Browser Native Event）封装为合成事件（SyntheticEvent）传入设置的事件处理器中。这里的合成事件提供了与原生事件相同的接口，不过它们屏蔽了底层浏览器的细节差异，保证了行为的一致性。另外有意思的是，React 并没有直接将事件附着到子元素上，而是以单一事件监听器的方式将所有的事件发送到顶层进行处理。这样 React 在更新 DOM 的时候就不需要考虑如何去处理附着在 DOM 上的事件监听器，最终达到优化性能的目的。

React 中调用 setState 之后发生了什么事情?

React 会将当前传入的参数对象与组件当前的状态合并,然后触发调和过程,在调和的过程中,React 会以相对高效的方式根据新的状态构建 React 元素树并且重新渲染整个 UI 界面.

React 得到的元素树之后,React 会自动计算出新的树与老的树的节点的差异,然后根据差异对界面进行最小化的渲染,在 React 的差异算法中,React 能够精确的知道在哪些位置发生看改变以及应该如何去改变,这样就保证了 UI 是按需更新的而不是重新渲染整个界面.

redux 中间件

中间件提供第三方插件的模式，自定义拦截 action -> reducer 的过程。变为 action -> middlewares -> reducer。这种机制可以让我们改变数据流，实现如异步 action ，action 过滤，日志输出，异常报告等功能。

常见的中间件：

redux-logger：提供日志输出

redux-thunk：处理异步操作

redux-promise：处理异步操作，actionCreator 的返回值是 promise

redux 有什么缺点

1.一个组件所需要的数据，必须由父组件传过来，而不能像 flux 中直接从 store 取。

2.当一个组件相关数据更新时，即使父组件不需要用到这个组件，父组件还是会重新 render，可能会有效率影响，或者需要写复杂的 shouldComponentUpdate 进行判断。

react 性能优化方案

重写 shouldComponentUpdate 来避免不必要的 dom 操作。

使用 production 版本的 react.js。

使用 key 来帮助 React 识别列表中所有子组件的最小变化。

什么时候在功能组件(Class Component)上使用类组件(Functional Component)?

如果您的组件具有状态(state)或生命周期方法，请使用 Class 组件。否则，使用功能组件