atitit.软件与sql与数据库设计模式大总结attialx总结v5 qc28.docx

[1.1. 版本历史 4](#_Toc343)

[2. 设计模式是什么 4](#_Toc28943)

[3. 软件设计模式起源于建筑领域，而建筑工程中设计模式的数量（253个） 5](#_Toc16107)

[4. 设计模式的结构 6](#_Toc24382)

[5. 软件模式并非仅限于设计模式，还包括架构模式、分析模式和过程模式 7](#_Toc32194)

[5.1. 设计模式与框架的区别 7](#_Toc3112)

[6. 具体的设计模式 7](#_Toc10705)

[6.1. Gof 23个经典设计模式(截止1994年） 7](#_Toc8758)

[7. 软件新发展的设计模式(1995-2016年）attilax总和大约80个.. 7](#_Toc276)

[7.1. Api映射法取代 warp法。 8](#_Toc6088)

[7.2. async异步模式 8](#_Toc22392)

[7.3. AOP 8](#_Toc29680)

[7.4. Abstract 优对于 interface 8](#_Toc5917)

[7.5. Ast模式 8](#_Toc25160)

[7.6. Auto type convert自动进行类型转换 8](#_Toc18274)

[7.7. biz业务化模块 8](#_Toc22064)

[7.8. Compiler as a Service 8](#_Toc31137)

[7.9. CBD（核心Core+行为Behavior+驱动Driver）架构模式 9](#_Toc16689)

[7.10. Chain mode 9](#_Toc12422)

[7.11. Confirm模式(解除页面跳转的问题) 9](#_Toc15716)

[7.12. closure闭包模式 9](#_Toc16230)

[7.13.](#_Toc21504) **[components组件化](#_Toc21504)** [9](#_Toc21504)

[7.14.](#_Toc18689) **[concurr并发模式(解除性能问题](#_Toc18689)** [9](#_Toc18689)

[7.15. DSL 9](#_Toc2438)

[7.16. Dwr vs AJAX 9](#_Toc396)

[7.17. DSLparser 9](#_Toc25638)

[7.18. eventDriver事件驱动模式 9](#_Toc13831)

[7.19. Code behind 9](#_Toc1604)

[7.20. Cfg2ui (attilax 创建 9](#_Toc27421)

[7.21. Biz\_index业务索引模式(attilax 创建,解 9](#_Toc1619)

[7.22. dispater分发器模式 9](#_Toc14367)

[7.23. declare声明式编程模式. 9](#_Toc13919)

[7.24. Dsl模式 9](#_Toc26341)

[7.25. filter过滤器模式 9](#_Toc24820)

[7.26. Fp模式 9](#_Toc26512)

[7.27. Engeen+script引擎+脚本模式(托管) 9](#_Toc28241)

[7.28.](#_Toc24452) **[frmwk框架化](#_Toc24452)** [9](#_Toc24452)

[7.29. 事件驱动 10](#_Toc15355)

[7.30. 事件分发 10](#_Toc5398)

[7.31. generic泛型模式 10](#_Toc562)

[7.32. -------------------h----n 10](#_Toc17857)

[7.33. Html5 ui 10](#_Toc15838)

[7.34. Hybrid化 10](#_Toc8922)

[7.35.](#_Toc2780) **[immediately](#_Toc2780)**[即时更新（热部署）快速迭代 持续交付 10](#_Toc2780)

[7.36. 10](#_Toc20769)

[7.37. Ioc/di 10](#_Toc31202)

[7.38. Lazy load延迟初始化 cfg 10](#_Toc27041)

[7.39. Linq 10](#_Toc1614)

[7.40. meta programe(anno,.net attr) 10](#_Toc11961)

[7.41. microcore微内核 10](#_Toc13209)

[7.42. msg消息化 11](#_Toc32324)

[7.43. Mvc mvp mvvm 11](#_Toc22350)

[7.44. ----o-------t 11](#_Toc4672)

[7.45. Orm 12](#_Toc31657)

[7.46. PageController 跟FrontController设计模式 12](#_Toc27488)

[7.47. Pipe 12](#_Toc26707)

[7.48. Pipe 管道模式、流式，链式 12](#_Toc23912)

[7.49. platform 12](#_Toc2512)

[7.50. plugin插件模式 12](#_Toc13134)

[7.51. Reflection 12](#_Toc30113)

[7.52. refresh刷新模式(解决缓存and静态的变量的值及时修改跟个读取问题) 12](#_Toc20846)

[7.53. refresh刷新按钮模式...解决若干页面不料条件..and 分页.. 12](#_Toc30757)

[7.54. propCopy数据转换器..propCopy 12](#_Toc11862)

[7.55. pool池模式 12](#_Toc17078)

[7.56. Retry(attilax 创建,提升稳定性) 12](#_Toc32496)

[7.57. Stream 13](#_Toc17313)

[7.58. Self contain自包含，容器化隔离。。 13](#_Toc6431)

[7.59. seri串口模式(attilax 创建,解除接口麻烦的) 13](#_Toc16101)

[7.60. spa ajax化 13](#_Toc26624)

[7.61. Try catch 13](#_Toc13789)

[7.62. 决缓存不及时的问题) 13](#_Toc26474)

[7.63. Theardlocal模式 13](#_Toc23873)

[7.64. trigger触发器模式(attilax 创建) 13](#_Toc2166)

[7.65. SOA 13](#_Toc7197)

[7.66. single单例文件夹模式(解决模块复制问题,提取问题) 13](#_Toc12032)

[7.67. 13](#_Toc9148)

[7.68. Timer 模式...解决同步问题两个控件 13](#_Toc14474)

[7.69. Theardlocal模式 14](#_Toc28558)

[7.70. ----u----z 14](#_Toc16105)

[7.71. uapi 14](#_Toc13712)

[7.72. Vm模式 14](#_Toc7586)

[7.73. visual 14](#_Toc21196)

[7.74. webService/HttpService 14](#_Toc14047)

[7.75. workflow工作流 14](#_Toc23774)

[7.76. Winform 14](#_Toc7528)

[7.77. Webform 14](#_Toc2215)

[7.78. Web page设计模式 14](#_Toc13059)

[7.79. Weak/Soft references 14](#_Toc22421)

[7.80. Promise设计模式 14](#_Toc1824)

[7.81. Yield模式 14](#_Toc18471)

[7.82. 管道模式，链式模式 14](#_Toc24526)

[7.83. Plugin设计模式 14](#_Toc27542)

[8. 反模式 14](#_Toc26425)

[8.1. Reset web 14](#_Toc4558)

[9. 参考资料 14](#_Toc2633)

软件的发展日新月异。。设计模式也是一直在发展。Gof 23个经典设计模式发表与1994年，也只记录了最常用的设计模式23个，但是，这个最常用的标准数量与内容因人而异和具体项目而不同，在当时1994年，设计模式的数量加上不常用的，按照20 80法则，应该总数也在100个左右。。

那么1995年以后，当然一直也有新的设计模式出现，只是没有经典书籍来总结了

软件新发展的设计模式(1995-2016年）attilax大概统计了下自身用到的设计模式大约80个

具体80个新总结的设计模式如下表所示。。

Mvc是不是设计模式，当然是设计模式来。。Struts，jsf，h5这类才是框架，才是实现mvc模式的代码实现。。..

## 版本历史

Frm atitit.软件设计模式大总结attialx总结v3 pa5

# 设计模式是什么

软件模式是在软件开发中某些可重现问题的一些有效解决方法

有若干通用性的solu

Design Pattern（设计模式）的目标是，把共通问题中的不变部分和变化部分分离出来。不变的部分，就构成了Design Pattern（设计模式）。这一点和Framework（框架）有些象。

#---如何抽取问题中的不变部分

下面举个排序的例子，说明如何抽取问题中的不变部分。

再次，按照field3的大小从小到大进行升序排序。

...

这种要求太多了，我们写了太多的重复代码。我们可以看到，问题的变化部分，只有判断条件部分（黑体的if条件判断语句）。

我们可以引入一个Comparator接口，把这个变化的部分抽取出来。

# 软件设计模式起源于建筑领域，而建筑工程中设计模式的数量（253个）

与很多软件工程技术一样，模式起源于建筑领域，毕竟与只有几十年历史的软件工程相比，已经拥有几千年沉淀的建筑工程有太多值得学习和借鉴的地方。

      那么模式是如何诞生的？让我们先来认识一个人——Christopher Alexander（克里斯托弗.亚历山大），哈佛大学建筑学博士、美国加州大学伯克利分校建筑学教授、加州大学伯克利分校环境结构研究所所长、美国艺术和科学院院士……头衔真多，IMG_256，不过他还有一个“昵称”——模式之父(The father of patterns)。Christopher Alexander博士及其研究团队用了约20年的时间，对住宅和周边环境进行了大量的调查研究和资料收集工作，**发现人们对舒适住宅和城市环境存在一些共同的认同规律**，Christopher Alexander在著作A Pattern Language: Towns, Buildings, Construction中把这些认同规律归纳为253个**模式**，对每一个模式(Pattern)都从Context（前提条件）、Theme或Problem（目标问题）、 Solution（解决方案）三个方面进行了描述，并给出了从用户需求分析到建筑环境结构设计直至经典实例的过程模型。

   1990年，软件工程界开始关注ChristopherAlexander等在这一住宅、公共建筑与城市规划领域的重大突破。最早将模式的思想引入软件工程方法学的是1991-1992年以“四人组(Gang of Four，简称GoF，分别是**Erich Gamma, Richard Helm, Ralph Johnson和John Vlissides**)”自称的四位著名软件工程学者，他们在1994年归纳发表了23种在软件开发中使用频率较高的设计模式，旨在用模式来统一沟通面向对象方法在分析、设计和实现间的鸿沟

# 设计模式的结构

应用场景and条件Context

Problem

解决

不适合领域

优缺点

  软件模式是在软件开发中某些可重现问题的一些有效解决方法，软件模式的基础结构主要由四部分构成，包括问题描述【待解决的问题是什么】、前提条件【在何种环境或约束条件下使用】、解法【如何解决】和效果【有哪些优缺点】，如图1-1所示：

**图1-1 软件模式基本结构**

    软件模式与具体的应用领域无关，也就是说无论你从事的是移动应用

# 软件模式并非仅限于设计模式，还包括架构模式、分析模式和过程模式

GoF将模式的概念引入软件工程领域，这标志着软件模式的诞生。软件模式(Software Patterns)是将模式的一般概念应用于软件开发领域，即软件开发的总体指导思路或参照样板。软件模式并非仅限于设计模式，还包括架构模式、分析模式和过程模式等，实际上，**在软件开发生命周期的每一个阶段都存在着一些被认同的模式**。

## 设计模式与框架的区别

设计模式是对在某种环境中反复出现的问题以及解决该问题的方案的描述，它比框架更抽象

# 具体的设计模式

## Gof 23个经典设计模式(截止1994年）

# 软件新发展的设计模式(1995-2016年）attilax总和大约80个..

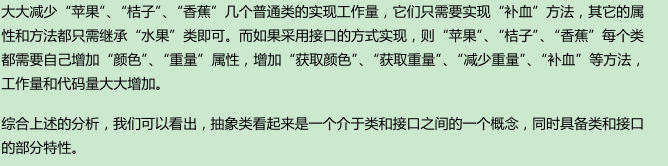
## Api映射法取代 warp法。

类似于hbuilder,将native api映射为我们的api，

## async异步模式

## AOP

## Abstract 优对于 interface



## Ast模式

## Auto type convert自动进行类型转换

Script

Jsp EL

Sql str>>num

## biz业务化模块

业务化的**业务模块，**框架，类库，模版，api

## Compiler as a Service

## ****CBD（核心Core+行为Behavior+驱动Driver）****架构模式

## Chain mode

## Confirm模式(解除页面跳转的问题)

## closure闭包模式

## ****components组件化****

## ****concurr并发模式(解除性能问题****

## DSL

## Dwr vs AJAX

## DSLparser

## eventDriver事件驱动模式

## Code behind

## Cfg2ui (attilax 创建

## Biz\_index业务索引模式(attilax 创建,解

## dispater分发器模式

## declare声明式编程模式.

## Dsl模式

## filter过滤器模式

## Fp模式

## Engeen+script引擎+脚本模式(托管)

## ****frmwk框架化****

## 事件驱动

## 事件分发

## generic泛型模式

## -------------------h----n

## Html5 ui

## Hybrid化

## **immediately** 即时更新（热部署）快速迭代 持续交付

（script+engine）模式

使用原生开发，从招聘、开发、上线各个环节的效率都慢一倍以上，而且参与的人越多，沟通效率往往拖慢不止一倍。

配置也应该使用dev模式,立即更新..可以..不需要重启app

## 

## Ioc/di

## Lazy load延迟初始化 cfg

## Linq

允许编写C#或者Visual Basic代码以查询数据库相同的方式操作内存数据。

## meta programe(anno,.net attr)

## microcore微内核

## msg消息化

这个图，是改用消息模式结果，用户注册功能模块，跟接收到消息处理各个任务直接，代码没有直接关联。如果需要增加一个接收注册消息，给其它人通知一下，只需要新增一个模块，订制注册消息，然后处理一个新流程即可。

目前各类开源项目里面，消息机制已经变得非常常见了。 但在wordpress这个blog框架，它的消息主题非常完善。几乎任何动作都有消息主题发送消息。 这样让开发者，只要订制相关主题，就可以，增加自己额外处理功能。 例如： 用户发一个帖子后。检测下用户是不是有广告信息，只要订制：发帖消息，然后，增加新功能，检测内容。发现不满足，直接屏蔽帖子。

有人估计要说，这类功能，现在很多框架有类似东西，例如，页面开始有个start事件，结束有个end事件。 确实，消息概念在很多框架里面确实有用。 但是，没有这么完毕的消息主题抛出。 **几乎做到整个系统任何操作，都能有消息，这样没有什么功能不能进行扩展了**。

## Mvc mvp mvvm

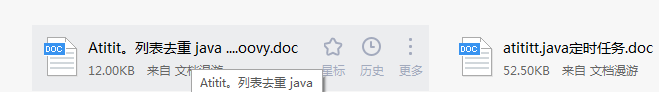
## ----o-------t

## Orm

## PageController 跟FrontController设计模式

## Pipe

## Pipe 管道模式、流式，链式



## platform

## plugin插件模式

## Reflection

## refresh刷新模式(解决缓存and静态的变量的值及时修改跟个读取问题)

## refresh刷新按钮模式...解决若干页面不料条件..and 分页..

## propCopy数据转换器..propCopy

## pool池模式

## Retry(attilax 创建,提升稳定性)

## Stream

## Self contain自包含，容器化隔离。。

iframe

## seri串口模式(attilax 创建,解除接口麻烦的)

## spa ajax化

## Try catch

## 决缓存不及时的问题)

## Theardlocal模式

## trigger触发器模式(attilax 创建)

## SOA

## single单例文件夹模式(解决模块复制问题,提取问题)

## 

## Timer 模式...解决同步问题两个控件

控件的onchange事件不能生效..

## Theardlocal模式

## ----u----z

## uapi

## Vm模式

## visual

## webService/HttpService

## workflow工作流

## Winform

## Webform

## Web page设计模式

## Weak/Soft references

## Promise设计模式

## Yield模式

## 管道模式，链式模式

## Plugin设计模式

# 反模式

## Reset web

# 参考资料

Atitit. 设计模式的总结Oa7.doc

atitit.软件设计模式大总结attialx总结v3 pa5.doc

atitit.设计模式的理解.txt

atitit.设计模式总结o5o

作者:: 绰号:老哇的爪子 （ 全名：：Attilax Akbar Al Rapanui 阿提拉克斯 阿克巴 阿尔 拉帕努伊 ）

汉字名：艾提拉（艾龙），   EMAIL:1466519819@qq.com

转载请注明来源： http://blog.csdn.net/attilax

Atiend