Atiitt 兼容性提升的艺术 attilax总结

目录

[1. 兼容性产生的原因 2](#_Toc11942)

[1.1. Api变化 2](#_Toc17395)

[1.2. 需求的资源不满足 2](#_Toc90)

[2. 兼容性的分类 2](#_Toc19692)

[2.1. Web方面的兼容性（js，html）,相同源码不同项目的实现 5 2](#_Toc15828)

[2.2. 8. 数据库表兼容性 7 2](#_Toc4979)

[2.3. 5. Api兼容性 3 2](#_Toc21760)

[2.4. 6. 接口兼容性 4 2](#_Toc22226)

[2.5. 9.2. 3.1. 完美的后向兼容性 vs 与前向兼容 2](#_Toc5344)

[3. 兼容级别 3](#_Toc2901)

[3.1. Abi兼容 3](#_Toc25732)

[3.2. Api兼容 3](#_Toc30721)

[4. 实现前向兼容 面向未来的兼容 3](#_Toc8803)

[4.1. 搜集新特性 未来基础版本api很可能实现这些新特性 3](#_Toc28386)

[4.2. 考虑草案api的兼容性 3](#_Toc30691)

[4.3. 多编码文字 3](#_Toc23826)

[5. 常见的出现兼容性的地方 4](#_Toc2525)

[5.1. 多字节编码的文字 ascii文字 4](#_Toc6051)

[5.2. 没有考虑汉字路径 空格路径等 4](#_Toc30286)

[5.3. 当前用户权限不足 4](#_Toc13575)

[5.4. 连续调用 4](#_Toc20639)

[6. 方法总结 4](#_Toc32444)

[6.1. @Deprecated模式 标注过期，但不删除 4](#_Toc6443)

[6.2. Try catch throwable 4](#_Toc9530)

[6.3. ,shim和polyfill.机制 4](#_Toc20968)

[6.4. 策略器模式 5](#_Toc28796)

[6.5. 适配器模式 5](#_Toc8541)

[6.6. 9.7. 6. 专门处理的软件列表 9 特殊判断 5](#_Toc14740)

[6.7. 谨慎修改 5](#_Toc15950)

[6.8. 冗余api备份机制，最起码同一个操作，使用俩个不同的api实现， 5](#_Toc7729)

[6.9. 兼容性api库 5](#_Toc29125)

[6.10. 增加新api说明 5](#_Toc9471)

[6.11. 新method加版本号数字日期版本号 6](#_Toc10847)

[6.12. 统一 api 多语言 6](#_Toc577)

[6.13. 使用中间库隔离层来兼容 6](#_Toc13064)

[6.14. 只新增，少修改，莫删除 2 6](#_Toc26142)

[6.15. 5.2. 增加新api 而不是 修改旧的api 4 6](#_Toc28809)

[6.16. 5.3. Threadlocal 参数兼容 6](#_Toc19062)

[6.17. 6.2. 修改接口（应该发布一个新接口，固化老接口 5 7](#_Toc25747)

[6.18. 8. 数据库表兼容性 7 7](#_Toc9275)

[6.19. 8.1. 2. 扩展表模式 7 7](#_Toc27191)

[6.20. 9. 兼容性策略 7 7](#_Toc15514)

[6.21. 兼容模式只针对开发者不愿意或者无法提供升级的场合 7](#_Toc32477)

[6.22. 兼容性补丁 8](#_Toc22657)

[7. 最佳实践总结 8](#_Toc17469)

[7.1. 兼容性文档与流程 8](#_Toc18756)

[7.2. 兼容性sdk库 8](#_Toc2796)

[8. Ref 8](#_Toc5995)

# 兼容性产生的原因

## Api变化

## 需求的资源不满足

比如新环境jar包不足

# 兼容性的分类

## Web方面的兼容性（js，html）,相同源码不同项目的实现 5

## 8. 数据库表兼容性 7

## 5. Api兼容性 3

## 6. 接口兼容性 4

## 9.2. 3.1. 完美的后向兼容性 vs 与前向兼容

# 兼容级别

## Abi兼容

## Api兼容

# 实现前向兼容 面向未来的兼容

## 搜集新特性 未来基础版本api很可能实现这些新特性

## 考虑草案api的兼容性

## 多编码文字

# 常见的出现兼容性的地方

## 多字节编码的文字 ascii文字

## 没有考虑汉字路径 空格路径等

## 当前用户权限不足

## 连续调用

# 方法总结

## @Deprecated模式 标注过期，但不删除

## Try catch throwable

## ,shim和polyfill.机制

在JavaScript的世界里,有两个词经常被提到,shim和polyfill.它们指的都是什么,又有什么区别?  
一个shim是一个库,它将一个新的API引入到一个旧的环境中,而且仅靠旧环境中已有的手段实现  
一个polyfill就是一个用在浏览器API上的shim.我们通常的做法是先检查当前浏览器是否支持某个API,如果不支持的话就加载对应的polyfill.然后新旧浏览器就都可以使用这个API了.术语polyfill来自于一个家装产品Polyfilla:  
  
Polyfilla是一个英国产品,在美国称之为Spackling Paste(译者注:刮墙的,在中国称为腻子).记住这一点就行:把旧的浏览器想象成为一面有了裂缝的墙.这些[polyfills]会帮助我们把这面墙的裂缝抹平,还我们一个更好的光滑的墙壁(浏览器)

shim 的概念要比 polyfill 更大一些，可以将 polyfill 理解为专门兼容浏览器 API 的 shim 。简单的说，如果浏览器X支持标准规定的功能，那么 polyfill 可以让浏览器 Y 的行为与浏览器 X 一样。

## 策略器模式

## 适配器模式

## 9.7. 6. 专门处理的软件列表 9 特殊判断

Jq就是这么搞得 提示兼容性

## 谨慎修改

## 冗余api备份机制，最起码同一个操作，使用俩个不同的api实现，

避免了某个api变动造成影响。。。最好把他搞成一个兼容性api库，方便使用。

## 兼容性api库

## 增加新api说明

在调用旧的api的时候，增加注释以及输出，表明此api已经过时，显示最新的api提示

## 新method加版本号数字日期版本号

## 统一 api 多语言

## 使用中间库隔离层来兼容

3. 3. 同时运行模式 1

3.1. 3.1. 完美的后向兼容性 2

3.2. 3.2. 虚拟机模式 2

3.3. 3.3. 版本兼容性模式 2

4. 4. 向前兼容(为升级预留足够余地） 3

5. 5. “向前兼容”理念 3

6. 6. 专门处理的软件列表 3

7. 新api vs 修改旧的api

## 只新增，少修改，莫删除 2

5. Api兼容性 3

5.1. 创建新的api应该使用什么标准??? 4

## 5.2. 增加新api 而不是 修改旧的api 4

## 5.3. Threadlocal 参数兼容

6. 接口兼容性 4

6.1. 新的接口去继承原来的接口，把新增的方法在子接口中声明。这样可以保持100%向前兼容 4

## 6.2. 修改接口（应该发布一个新接口，固化老接口 5

7. Web方面的兼容性（js，html）,相同源码不同项目的实现 5

7.1. 15.1. 不同的项目与不同的实例启动 5

7.2. 15.2. 不同的项目与实例配置文件 6

7.3. 15.3. Web.xml怎么办？？ 6

7.4. 15.4. 跳转到同一功能spec的不同实现 6

7.5. 15.5. 不同项目的同一功能就实现可以放在同一上级模块package下 6

## 8. 数据库表兼容性 7

## 8.1. 2. 扩展表模式 7

## 9. 兼容性策略 7

9.1. 3. 同时运行模式 7

9.2. 3.1. 完美的后向兼容性 8

9.3. 3.2. 虚拟机模式 8

9.4. 3.3. 版本兼容性模式 8

9.5. 4. 向前兼容(为升级预留足够余地） 9

9.6. 5. “向前兼容”理念 9

9.8. Atitit.兼容性的“一加三”策略（不推荐） 10

## 兼容模式只针对开发者不愿意或者无法提供升级的场合

对于销量很大而用户忠诚度很高的软件中的影响用户升级的软件bug，微软提供兼容模式，模拟旧版本的Windows的部分行为，作为面向最终用户的临时解决方案。但是微软不可能一直这样惯着有bug的程序，这对提高整个生态环境中的软件质量没有帮助，比如不必要地要求管理员权限的程序造成用户较难不用管理员身份登录系统，增加用户被攻击的风险

，而且并不能解决所有的兼容性问题。

## 兼容性补丁

# 最佳实践总结

## 兼容性文档与流程

## 兼容性sdk库

# Ref

Atitit.软件兼容性原理与实践 v5 qa2.docx

atitit 软件架构高扩展性and兼容性原理与概论实践attilax总结 v2q0f.docx

Atitit.提升api兼容性的方法 v2 q326.docx

jq 如何解决兼容性的

JavaScript术语：shim和polyfill\_搜狐科技\_搜狐网.html