



# Agenda

- AIR Performance
- AIR Security
- Q & A



# Part 1

# 创建出色的AIR应用

# Outline

## 此次演讲不仅仅是:

- 一堆关于技术工作的说明
- 关于AIR应用的说明

## 此次演讲是:

- 能够帮助你优化任何AIR应用
- 演示通用工具和优化技巧
- 演示AIR特定的工具和优化技巧

# What is performance?

## 有关性能方面狭隘的问题：

- 我的应用能跑的很快么？
- AIR在编写实际应用方面足够快么？
- 做某某X任务AIR是不是很慢？

# What is performance?

## 关于性能更好的问题：

- 哪些计算操作对于性能要求更加敏感？
- 有没有我能用来衡量这种敏感要求的实际数据？
- 根据数据我如何能够调优我的应用？

# Performance-Sensitive Operations

## 导致性能敏感操作的范例:

- 巨大显著的处理 (image, sound, video)
- 渲染 (大型datasets, 3D)
- 搜索
- 从服务器获取数据
- 响应用户输入交互

# Defining Metrics

## 好的数据:

- **可以计算的**—你能够计算产生的数据
- **连续的, 一贯的**—当不断重复测量时
- **有意义的**—与你所关心的信息相关联

# An Example

假设:

- 向一个100MB大小的图片应用滤镜
- 显示进度并且允许用户取消

数据:

- 吞吐率(bytes/msec)
- 内存占用(bytes)
- 响应时间(msec)

# Taking Measurements

## 衡量吞吐率:

- `start_msec = new Date().time`
- `filter();`
- `end_msec = new Date().time`
- `rate = imageSize_bytes / ( end_msec - start_msec )`

# Taking Measurements

## 衡量内存占用:

- 使用监控工具, 比如Mac上的Activity Monitor或Win上的Process Explorer
- 衡量驻留内存尺寸或私有工作集合尺寸

WS private bytes 是OS上的RAM数量, 是用于专门保持应用程序独立运行的内存大小

# Taking Measurements

## 响应时间:

- 开始计时, 从用户开始操作时
- 等待应用响应
- 停止及时, 当UI显示结果时
- 延时超过100毫秒或能够直接感觉出来延时

# Optimization Process

## 重复操作直至满意:

- 衡量
- 分析
- 修改

## 宽泛地讨论修改方面:

- 你的程序设计(高影响度)
- 代码本身(低影响度)

# Using Test Applications

## 考虑同完成目标构建的测试应用一起工作

- 全部应用很复杂或者很难被理解
- 应用的大部分都不需要被优化
- 保证测试工具独立于你的应用
- 你需要重新验证变更过的应用

# Demonstration

# Chunking Work—Pseudocode

```
var timer:Timer = new Timer( 1, 1 );  
  
...  
  
var start:Number = new Date().time;  
while( workRemaining ) {  
    doWork();  
  
    var now:Number = new Date().time();  
    if(( now - start ) > TIME_LIMIT ) {  
        if( workRemaining )  
            timer.start();  
        break;  
    }  
}
```

# Optimization is a Process

## 你怎样针对可能的变化进行优化

- 新的功能可能导致完全不同的重要数据结果
- 用户的期望仍然要满足

## 优化可能带来的副作用

- 你需要在各种数据结果上进行均横评测

## 优化可能带来的影响

- 编译器的变动可能使实现更加迅速
- 库和代码依赖关系上的改变可能变更他们的特征
- 硬件上的变化

# Examples of news affecting the game

Flash Player 引入Pixel Bender, 3D变形等功能, 这些可能提升应用的性能

## **Pixel Bender:**

- 使用高效率的编程语言用来并行处理像素上的操作
- 在Flash Player中仅利用CPU, 还没有GPU计算部分
- 在多个CPU内核上的并行计算可行

你能够使用它在大量的数字处理上, 你能够通过对于ByteArrays或Vectors的操作来替代对于bitmaps整体的操作:

- 混合声音, 处理大型数据集合
- 使用这种模式, 将使用独立于AS线程的独立线程进行运算, 你将获取一个自由的UI响应线程

## Keep learning

举例，如果你的应用是操作mp3文件，你可能知道一个很有趣的事情，Flash Player内部操作这些文件是用44khz的采样率来工作的

因此，如果你提供了其他采样率的文件，那么Flash Player就要花费额外的CPU进行重新采样

# Tips

- 变更默认的FPS, 从24到尽可能最小(保证动画平滑), 当应用闲置时, FPS改设定为
- 不要在ENTER\_FRAME事件上用侦听器, 用Timers代替
- 总是使用unloadAndStop命令把不需要的内容卸载掉
- 不要Timers时候, 要stop
- 在摧毁一个对象实例前先卸载相关事件侦听器
- 使用正确的数据类型, 比如你不需要数据的绑定功能, 就尽量用Array而不是ArrayCollection, 如果你有一个滤镜功能的方法, 在你添加一个UI新元素前, 没必要调用refresh()之类的方法
- 更倾向于使用大的对象而不是无数个小的对象来解决逻辑问题, 这会避免内存片段的过度分离
- 尽可能的重用对象, 无论任何时候, 它总要比重新创建实例, 构造函数和初始化流程更加节省内存, 时间和CPU
- 尽可能使用内置常量帮助编程, 使用Object替代无类型或无确认类型的数据。

# Some resources

Oliver Goldman's blog:

<http://blogs.adobe.com/simplicity/>

Andrew Trice's 10 Tips for Flex Application Performance:

<http://www.insideria.com/2009/09/10-tips-for-flex-application-p.html>

Arno Gourdol's Writing well-behaved, efficient, AIR applications:

<http://arno.org/arnotify/2009/05/writing-well-behaved-efficient-air-applications/>

About Pixel Bender (Tinic Uro's blog):

<http://www.kaourantin.net/2008/05/adobe-pixel-bender-in-flash-player-10.html>

Examples of using Pixel Bender in Flex (also applied on sounds; Miti's blog):

<http://miti.pricope.com/2008/11/10/playing-with-pixel-bender/>

Sean Christmann's Optimizing Adobe AIR for Code Execution, Memory, and Rendering:

<http://tv.adobe.com/#vi+f15384v1018>

<http://www.craftymind.com/2008/11/20/max-2008-session-material/>



# Part 2

# AIR 安全性

# Outline

- 应用签名
- 更新框架
- 本地存储数据
- AIR 安全沙箱
- 验证数据

## 今天的演讲关于：

- 让应用更加安全和避免攻击的技巧
- 通过应用加密安全存储用户数据的技巧
- 没有关于DRM的话题

# What is AIR

- 概念上类似于EXEs可执行文件
- 数字签名和用户批准的安装过程
- 通过HTML/JS或ActionScript创建
- 跨平台: Windows, Mac 和Linux
- 完全访问桌面
- 在用户许可下运行

# Why security is so important?

当一个Web应用被危及时，通常都是服务器端数据被泄露

当一个桌面应用被危及时，攻击者有机会获取整台机器的控制权

Adobe AIR提供一系列的工具有帮助开发人员实现安全的应用

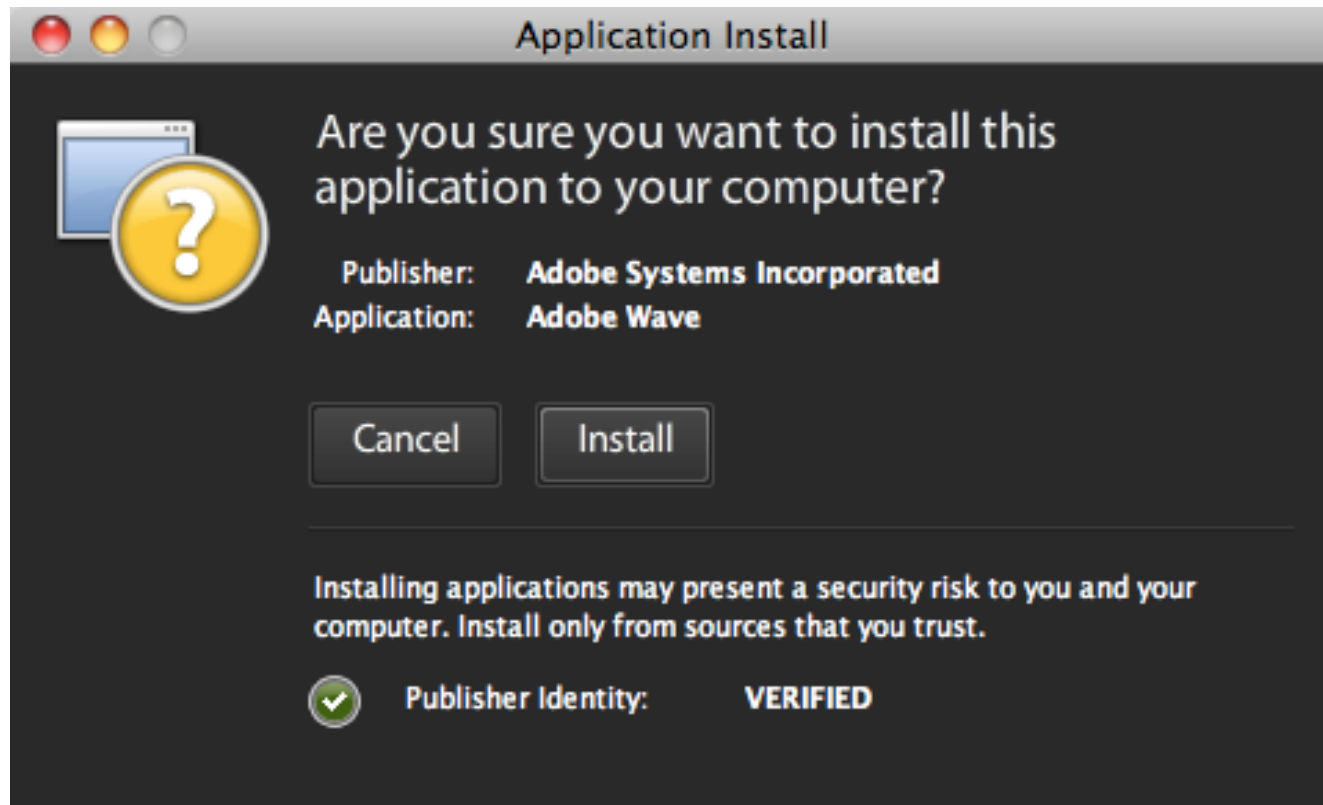
# Application signing

- 如何签名?
  - 从已经授权的安全验证处取得一个可用的数字签名
- 签名验证的类型?
  - AIR需要签名验证应用代码的设计特征
- 能做什么?
  - 验证应用程序源的保全 (应用发布者的确是唯一经过验证的人)
  - 确保应用没有被篡改 (经由Server端到客户端的传输途径)
- 到哪里购买?
  - 任意的安全数字签名机构的验证均可购买使用

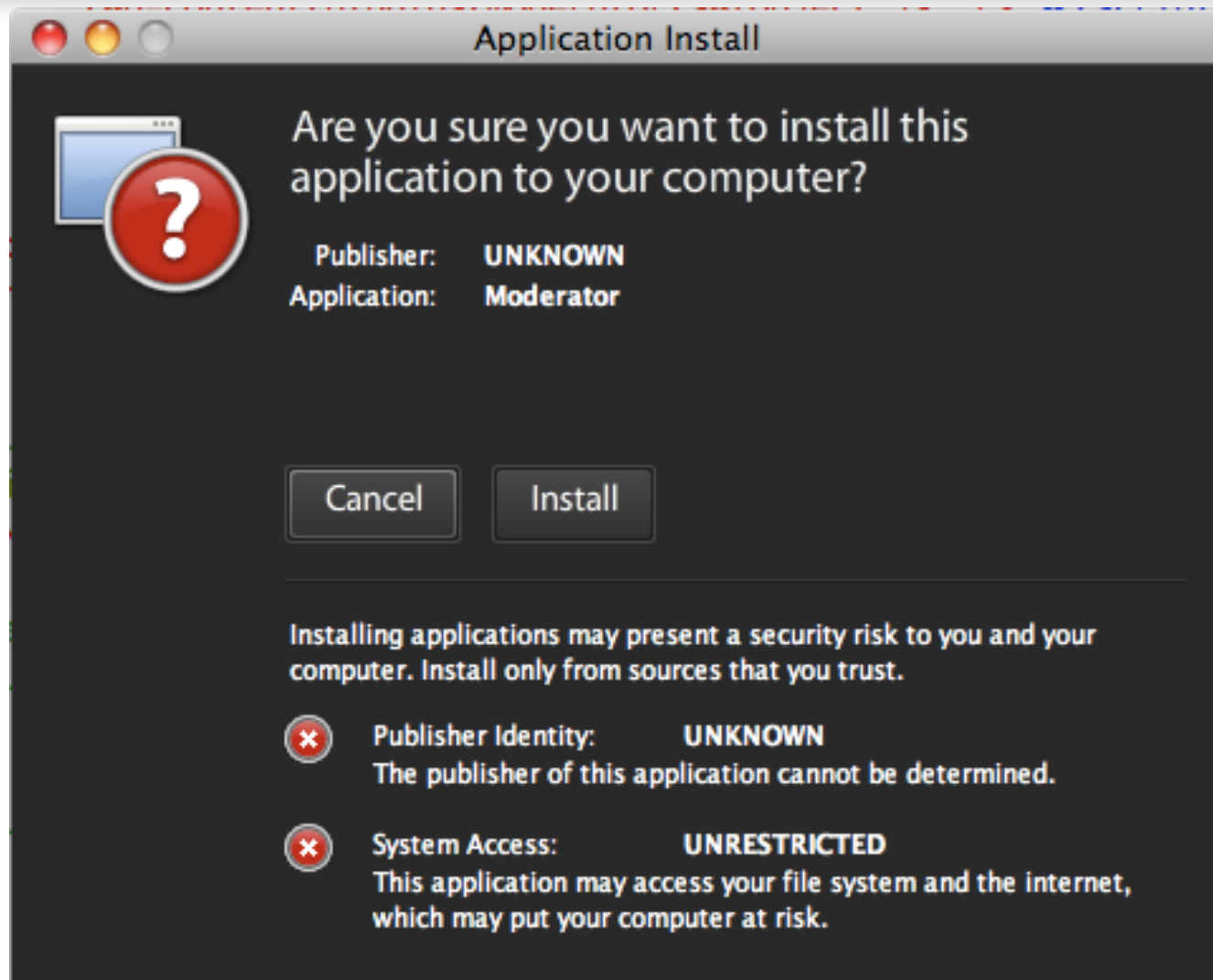
# Application signing

- Adobe AIR不提供任何工具来直接管理可供信任的签名文件
- 管理信任的签名文件的设备已经由操作系统提供, 通过这些工具可以添加和删除授信签名证书文件
- Adobe AIR认为一个证书是可信任的
  - 存储于系统中的信任证书与作为其可用的信任证书是相同的一个文件
  - 或者如果签名证书从获取签名验证的体系或机构中建立一个可供信任的关系的文件

# Application signed with a trusted certificate



# Application signed with self generated certificate



# Workflows for signing an application

有两种不同的workflows能够支持通过Flex Builder或者ADT的签名

1. 当输出release版本时, 可以从Flex Builder中签名
2. 或者你的程序员可以输出一个立即文件格式(AIRI):
  - 这个文件不能被安装
  - 使用这个文件, 拥有这个认证的人, 可以使用ADT命令行工具直接生成签名

# Migrate your app from one certificate to another

你能够改变签名验证, 从一个已经存在的验证, 变更为一个新的验证签名, 在旧的签名还仍然有效的条件下:

1. 应用使用新的验证签名
2. 使用ADT工具, 配合命令去签署一个有旧验证签名的.air文件

因此你能够:

- 将一个单方自验证签名移植到经由CA发行的可信签名之上
- 从一个商业签名变更为另外一个

签名的变更可以工作于安装过程或应用更新过程

## More on signing an application

- 你不能够对一个过期的或者是废弃的验证进行签名
- 你能够对一个可续签的验证进行签名
- 你能够安装一个已经有可用签名验证的AIR应用，甚至是一个已经过期的签名的AIR应用

# Application Identity

- AIR使用签名证书建立起应用的信用体系
- 应用本体通过安全的方式验证以下操作：
  - 更新应用时
  - 浏览器API被用于安装, 检测, 从浏览器内启动应用时
  - 与其他AIR或网络应用通过LocalConnection方式通讯时
  - 验证当一个应用访问加密的本地数据库(ELS)时

# Updating AIR applications

AIR 1.5提供了一个更新框架，用于安全的更新一个已经安装的AIR应用:

- 仅通过应用描述配置文件中的版本号的不同，是不能更新一个已经安装的AIR应用的(不安全)
- 一个有签名证书的应用，不能被随意的其他版本的应用更新，除非使用了同一个签名证书。使用命令行的本地打包更新除外
- 即便更新版本使用了一个被续签的安全证书，发布人的ID也应该相同才可以更新

# Updating AIR apps with Update Framework

你能够使用下列流程轻松利用更新框架:

- 每次应用启动时, 首先检查是否有新的更新
- 如果一个更新被发现, 你可以选择询问用户是否更新或者自动先行下载更新
- 一旦更新被下载, 被安装, 应用会自动重新启动

使用这种更新流程, 用户只要在线, 就会有选择的进行更新或者不进行更新  
方式很灵活, 你可以随意定制它

# Updating AIR apps with Update Framework

来自于2种方式:

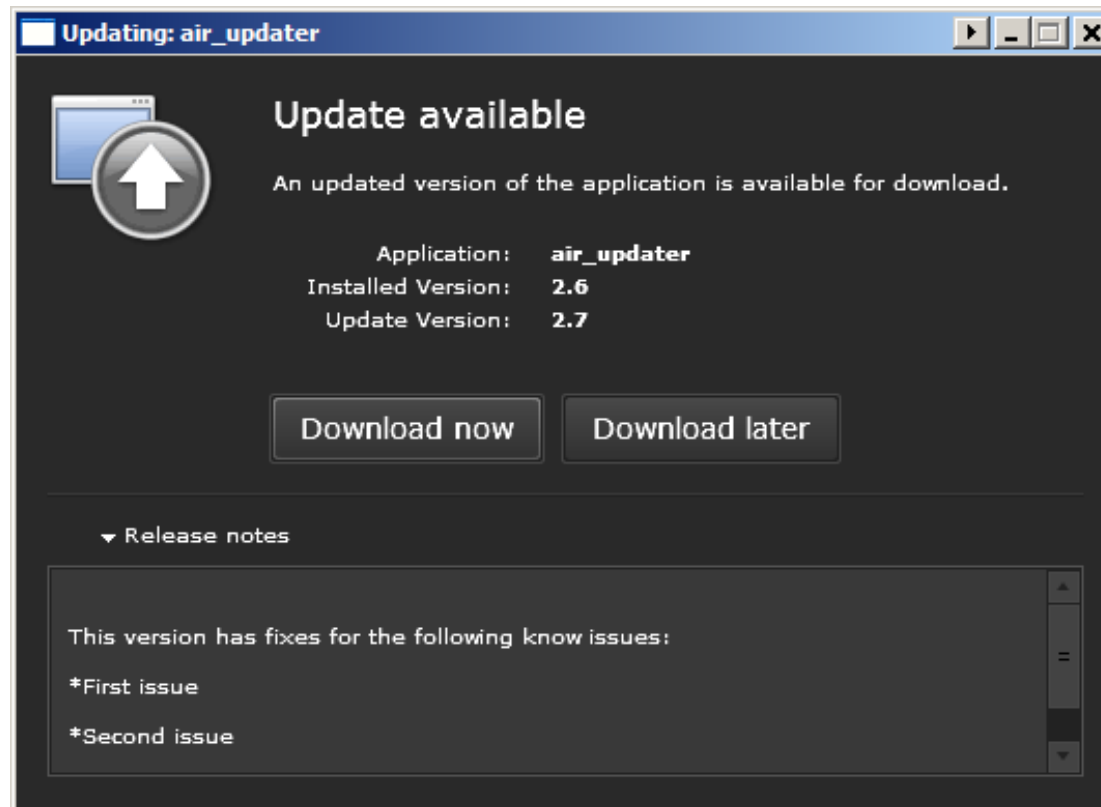
- 使用框架UI
- 不使用框架UI, 你必须自己实现UI

AIR应用目前可以通过以下方式创建:

- Flex (2种)
- AJAX (2种)
- Flash (自己实现更新UI)

# Updating AIR apps with Update Framework

你可以为用户显示一个描述更新原因的对话框体:



# Storing data locally

AIR提供不同的方式将数据存储在客户端

1. 保存文件到系统本地
2. 序列化ActionScript对象到本地系统文件中
3. 存储ActionScript对象到加密本地存储
4. 存储数据, 使用本地SQLite数据库

## Storing data locally

前两个存储方式不提供任何加密安全保护，任何其他的应用都可以读取存储的数据。

你应该使用app-storage:/文件目录，因此你能够保存每个用户帐户对应的数据。如果相同的应用在一台机器上被多个用户帐户使用，那么也不会影响其中的数据

不要在app:/文件夹内创建或修改文件

# Encrypted Local Store

加密本地存储, 提供一系列安全功能:

- AIR使用DPAPI在Windows上, KeyChain钥匙链在Mac OS上来关联每个应用和用户之间的加密本地存储。加密本地存储使用128位AES加密算法
- 很适用于小型数据, 每个应用空间不大于10MB, 性能限定
- 内容只有代码从应用安全沙箱许可范围内运行才变的可以访问
- 默认情况下, 数据绑定到发布者ID, 你也能够绑定到应用本身, 那么应用一旦更新发生变动, 此部分数据将不再可用
- 一个ELS对应一个应用和用户帐户

`EncryptedLocalStore.setItem(key:String, data:ByteArray, stronglybound:Boolean)`

# Encrypted Local Databases

- SQLite存储整个数据库到一个独立文件，并且这个文件会被加密
- 你使用一个16字节长度的Key来加密和解密数据库文件
  - 可以使用SHA2-256来生成Key，通过as3corelib库中的
- 一个数据库在创建时选择不加密，那么将不能在以后改为加密，算法不一致

```
sqlConnection = new SqlConnection();
```

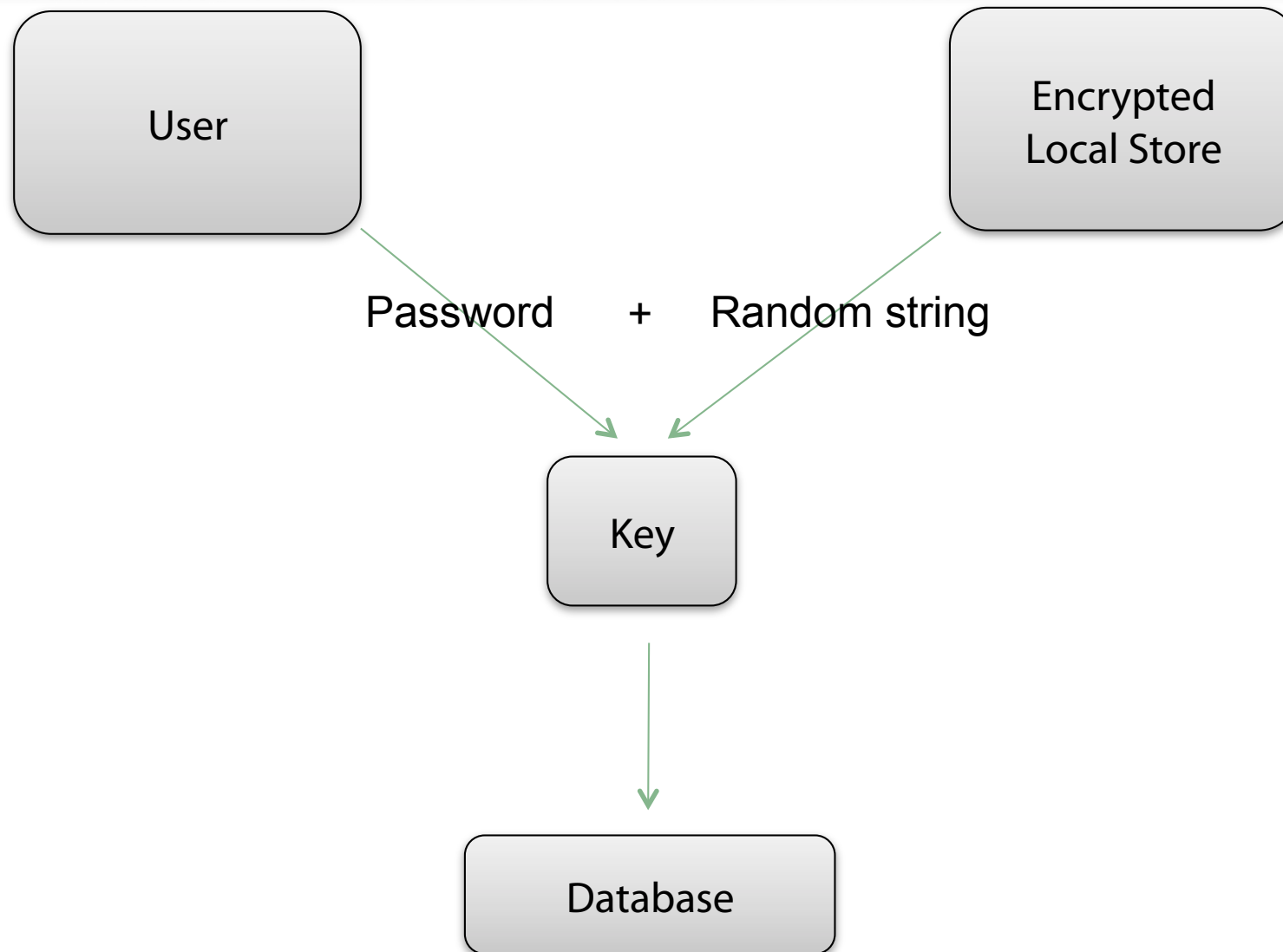
```
sqlConnection .open(reference:Object = null, openMode:String = "create",  
    autoCompact:Boolean = false, pageSize:int = 1024, encryptionKey:ByteArray = null):
```

# Encrypted Local Databases

需要牢记:

- 加密数据库不能抵御SQL注入式攻击
  - 数据库的安全依赖于如何保护Key的安全, AIR应用能够被反编译, 所以不要将Key串硬编码到程序中
  - 针对Key你可以使用用户密码来获取, 或者生成一个随机数字与用户密码来组合获得
  - 你能够使用ELS来保存Key或者Key的一部分
  - 你能够使用SHA256.hash() 来哈希处理密码, 该功能在as3Corelib中
- 看看BlackBookSafe 如何处理.

# Encrypted Local Databases



有2种安全沙箱, 从安全的角度来看代码如何运行:

1. 应用安全沙箱Application sandbox – 允许访问有授权的AIR APIs(read/write OS files...).
  - 任何来自于AIR安装包内的文件都置于此安全沙箱中
2. 非应用安全沙箱Non-application sandbox – 这里的内容没有权限访问AIR APIs
  - 任何从本地或远程读取加载的内容都是非应用安全沙箱

## More on SandBoxes

你能够在在一个非应用安全沙箱的特别安全级别下，从应用安装目录下运行一段代码

使用*sandboxRoot* 和*documentRoot*

# Communicate between Sandboxes

你能够使用SandboxBridge API在两个沙箱间通讯.

- 任何数据被传送, 都是以结果值的形式, 而不是对象引用, 这里不会有任何引用泄露
- 你不能暴露通用函数给非应用沙箱:  
**deleteConfigFile()** 代替**deleteFile(fileToDelete:String)**
- 没有模棱两可的安全, 任何方法通过SandboxBridge暴露的方法都能够在另一端被发现
- 选择在两边

# Imported vs. Sandboxed

你可以读取可执行文件 (SWF, HTML/JS) :

1. Sandboxed: 保留在其自己的域名内;
  1. HTML: <frame>, <iframe>, <img> tags
  2. SWF: Loader.load(), SWFLoader
2. Imported: 运行loader的沙箱和loader的授权权限
  1. HTML/JS: eval(), innerHTML, <script>
  2. Loader.loadBytes(), HTMLLoader.loadString()

当你导入一个外部内让那个到应用沙箱时, 你就给这个内容赋予了AIR API的访问权限

# AIR Prevents Accidental Importing

你能够读取可执行文件 You can load executable content (SWF, HTML/JS):

- 在HTML中, In HTML, eval() 是被禁止的.
  - 在onLoad前, 他们都操作正常
  - 在onLoad后, 他们不能再产生代码
  - 限制eval()支持纯粹的JSON, 但是是一些系统还是允许生成代码.
- Loader.loadBytes() 和HTMLLoader.loadString() 需要选择
  - LoaderContext.allowLoadBytesCodeExecution (只能是AIR)
  - HTMLLoader.allowLoadStringInAppSandbox (NEW in AIR 1.5!!!)

# Verify Imports

特别当你导入到应用安全沙箱时，你应该验证导入的代码是你想要的代码  
最安全的方式，是使用代码签名

坏消息: 没有任何内置到运行时和SDK的支持能够轻松验证

好消息: Alchemy能够把C/C++库移植到ActionScript中，以此编写自己的框架  
来验证内

服务器指定和服务器+SSL还不够完善，你还可能遇到DNS攻击，服务器内容  
攻击，代理攻击等威胁

# Data Validation

不论你做什么在网络上，你都要在AIR里做，总是验证数据，不要全部完全信任客户端

使用Flex校验控件，Flash校验控件 (Google Code Project)

SQL 注入:当写数据给SQLite时，使用prepare 声明

不用:

```
employees.text = "SELECT FROM employees WHERE employeeID = " + remotedata.ID;  
employees.execute();
```

用:

```
employees.text = "SELECT FROM employees WHERE employeeID = :empID";  
employees.parameters[":empID"] = remotedata.ID;  
employees.execute();
```

# Articles

<http://ddj.com/web-development/210004209>

[http://www.adobe.com/devnet/air/ajax/articles/blackbooksafe\\_anatomy.html](http://www.adobe.com/devnet/air/ajax/articles/blackbooksafe_anatomy.html)

[http://www.adobe.com/devnet/air/articles/air\\_update\\_framework.html](http://www.adobe.com/devnet/air/articles/air_update_framework.html)

[http://www.adobe.com/devnet/air/articles/introduction\\_to\\_air\\_security.html](http://www.adobe.com/devnet/air/articles/introduction_to_air_security.html)

[http://www.adobe.com/devnet/flashplayer/articles/secure\\_swf\\_apps\\_print.html](http://www.adobe.com/devnet/flashplayer/articles/secure_swf_apps_print.html)

<http://onair.adobe.com/blogs/videos/2008/06/23/ethan-malasky-developing-secure-air-applications/>

[http://weblogs.macromedia.com/emalasky/archives/2008/04/remote\\_plugins.html](http://weblogs.macromedia.com/emalasky/archives/2008/04/remote_plugins.html)

[http://www.adobe.com/devnet/air/flex/quickstart/xml\\_signatures.html](http://www.adobe.com/devnet/air/flex/quickstart/xml_signatures.html)



# Q & A

Thank You!

[mihai.corlan@adobe.com](mailto:mihai.corlan@adobe.com)

<http://corlan.org>





**Adobe**