

# 持续集成在Flex/AS3开发中的应用

JexChan @gmail.com

新浪微博: <http://weibo.com/agilelife>

持续集成简介

持续集成实践

扩展阅读

Q & A

# 一个简单的调查...

Martin Fowler  
对持续集成的定义

持续集成是**一种软件开发实践**，团队开发成员经常集成他们的工作，每天可能会发生多次集成。每次集成都通过**自动化的构建**（包括编译，发布，自动化测试）来验证，从而**尽快地发现集成错误**。许多团队发现这个过程可以大大减少集成的问题，让团队能够更快的开发内聚的软件。

<http://martinfowler.com/articles/continuousIntegration.html>

# 持续集成的关键是...

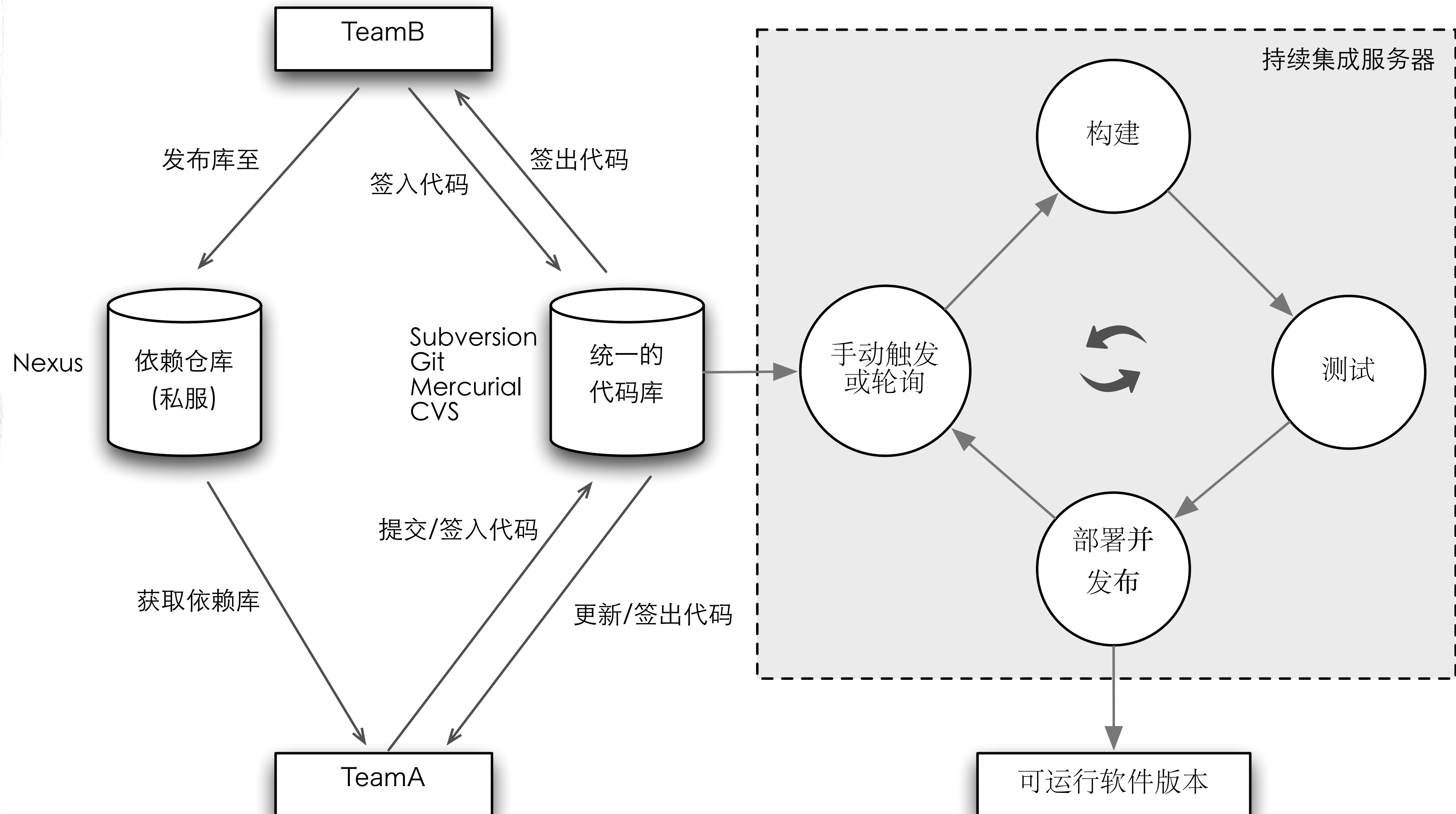
快速反馈

自动化构建

软件开发实践而不是工具



# 持续集成实践流程示意



## 持续集成工具箱



1. 使用Ant构建Flex应用程序
2. 应用Ivy配合Ant进行依赖管理
3. 使用Nexus作为版本库仓库
4. 使用Hudson/Jenkins作为持续集成服务器



# 1 使用Ant进行Flex应用程序构建

flexTasks.jar

Flash Builder安装目录\sdk\4.5.0\ant\lib\

flexunitTasks-4.1.0-8.jar

在下载FlexUnit压缩包中可以找到

build.xml

build.xml

```
<!-- 加入构建Flex应用程序需要的Ant扩展包flexTasks.jar -->  
<taskdef resource="flexTasks.tasks"  
classpath="${FLEX_HOME}/ant/lib/flexTasks.jar" />
```

```
<!-- 加入运行单元测试需要的Ant扩展包flexUnitTasks.jar -->  
<taskdef resource="flexUnitTasks.tasks" >  
  <classpath>  
    <fileset dir="${lib.loc}">  
      <include name="flexUnitTasks*.jar" />  
    </fileset>  
  </classpath>  
</taskdef>
```

# 常用的Ant任务

**<mxmllc>** 编译.MXML文件

**<compc>** 编译.AS文件

**<html-wrapper>** 生成SWF文件的HTML包装页面

**<asdoc>** 生成ASDoc文档

---

**<flexunit>** 运行单元测试并以XML形式生成测试报告

**<junitreport>** 根据上述XML生成HTML测试报告页面



## 编译主程应用程序

```
<!-- 编译主程序 -->  
<target name="compile" depends="init">  
  <mxmhc file="${src.loc}/FlexAntFirstStep.mxml"  
    output="${build.loc}/FlexAntFirstStep.swf" />  
</target>
```

ant compile

## 生成HTML封装页面

```
<!-- 使用HTML页面来包装生成的主程序 -->  
<target name="package" depends="compile">  
  <html-wrapper swf="FlexAntFirstStep"  
    output="${build.loc}"  
    height="100%" width="100%" />  
</target>
```

# ant package

## 生成ASDoc文档

```
<!-- 生成ASDoc文档 -->  
<target name="build-doc" depends="init">  
  <asdoc output="${build.doc}" failonerror="true"  
    main-title="Created with Ant"  
    footer="Copyright 2011 jchen">  
    <doc-sources path-element="${src.loc}/com/jchen/events"/>  
  </asdoc>  
</target>
```

# ant build-doc

## 生成测试报告XML文件

```
<!-- 以XML形式输出测试运行报告 -->  
<target name="test" depends="compile">  
  <flexunit swf="${dist.loc}/TestRunner.swf"  
toDir="${report.loc}" haltonfailure="false" />  
</target>
```

ant test



## 生成测试报告HTML页面

```
<!-- 根据test阶段生成的XML文件来生成测试运行报告的HTML页面 -->  
<target name="report" depends="test">  
  <junitreport todir="${report.loc}">  
    <fileset dir="${report.loc}">  
      <include name="TEST-*.xml" />  
    </fileset>  
    <report format="frames" todir="${report.loc}/html" />  
  </junitreport>  
</target>
```

ant report



操作演示...

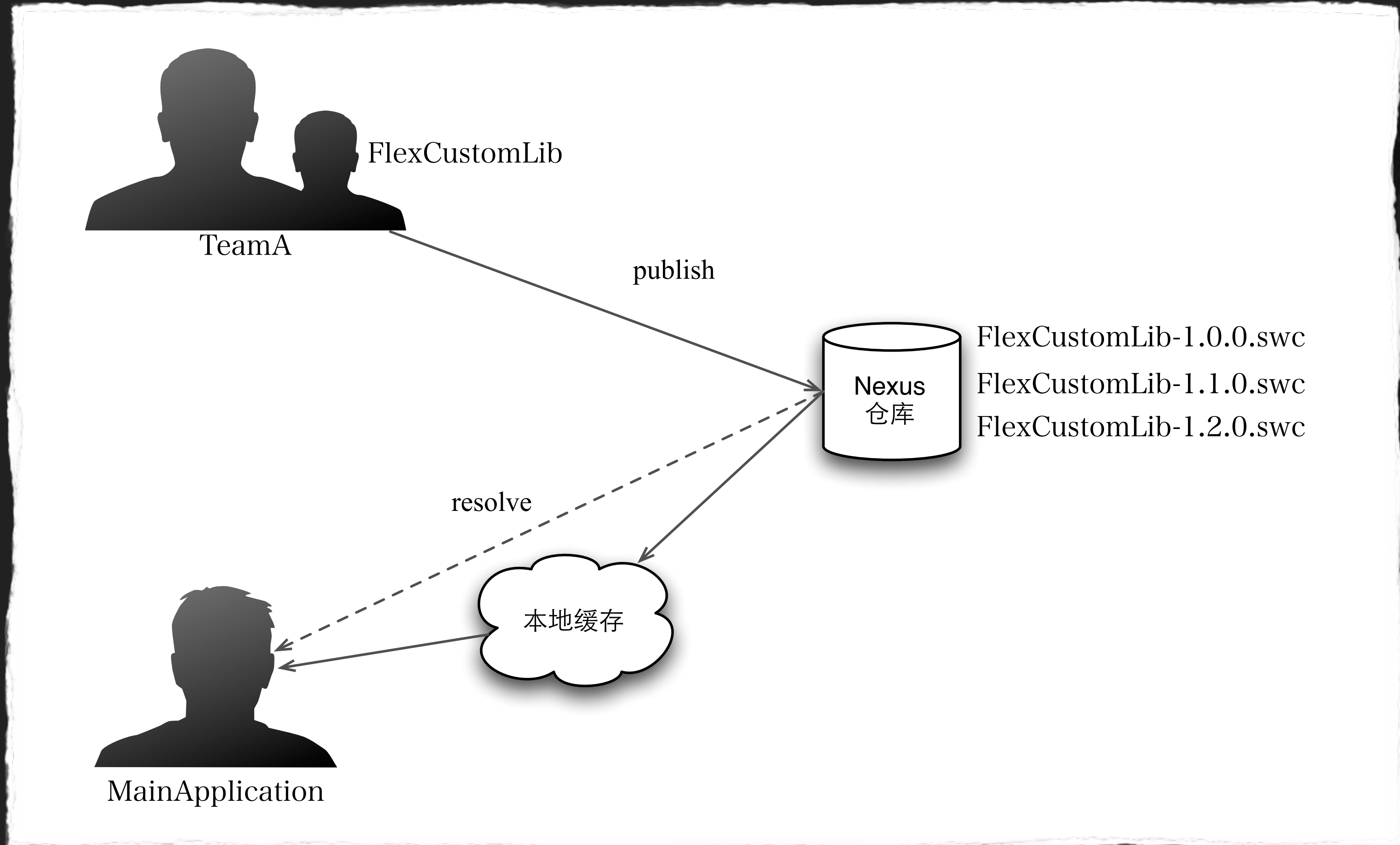
# 小技巧:

Flash Builder安装目录\sdk\4.5.0\bin\

**mxmcl -help list advanced details**

查看命令参数及用法

## 2 应用Ivy配合Ant进行依赖管理



**ivy.xml** 依赖关系配置与设定

**ivysettings.xml** 仓库配置及路径匹配模式设置

**build.xml** 调用Ivy相关的任务完成依赖关系解决

## build.xml 导入ivy命名空间

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>  
<project name="FlexCustomLibs"  
  xmlns:ivy="antlib:org.apache.ivy.ant">  
</project>
```



## 3 搭建Nexus版本仓库(私服)


<http://nexus.sonatype.org/downloads>

将war包部署到常见的Web容器中 (nexus-webapp-1.9.2.3.war)

[←](#)
[→](#)
[↻](#)
[🏠](#)

localhost:8080/nexus/index.html#view-repositories

☆ 🔍



# Sonatype

Log In

Sonatype Nexus™ Open Source Edition, Version: 1.9.2.3

Sonatype™ Servers

Nexus

Artifact Search

Advanced Search

Views/Repositories

Repositories

Help

Welcome

Repositories

Refresh

User Managed Repositories

Repository	Type	Format	Policy	Repository...	Repository Path
<b>Public Repositories</b>	group	maven2			http://localhost:8080/nexus/content/groups/public
3rd party	hosted	maven2	Release	In Service	http://localhost:8080/nexus/content/repositories/thirdparty
Apache Snapshots	proxy	maven2	Snapshot	In Service	http://localhost:8080/nexus/content/repositories/apache-snapshots
Central M1 shadow	virtual	maven1	Release	In Service	http://localhost:8080/nexus/content/shadows/central-m1
Codehaus Snapshots	proxy	maven2	Snapshot	In Service	http://localhost:8080/nexus/content/repositories/codehaus-snapshots
Google Code	proxy	maven2	Release	In Service	http://localhost:8080/nexus/content/repositories/google
java.net - Maven 2	proxy	maven2	Release	In Service	http://localhost:8080/nexus/content/repositories/java.net-m2
java.net-m1	proxy	maven1	Release	In Service	http://localhost:8080/nexus/content/repositories/java.net-m1
java.net-m1 M2 shadow	virtual	maven2	Release	In Service	http://localhost:8080/nexus/content/shadows/java.net-m1-m2
Maven Central	proxy	maven2	Release	In Service	http://localhost:8080/nexus/content/repositories/central
Releases	hosted	maven2	Release	In Service	http://localhost:8080/nexus/content/repositories/releases
Snapshots	hosted	maven2	Snapshot	In Service	http://localhost:8080/nexus/content/repositories/snapshots

Select a record to view the details.

操作演示...

## 4 搭建Hudson持续集成服务器

<http://java.net/downloads/hudson/war/>

也可以将war包部署到常见的Web容器中 (hudson-2.2.0.war)

```
java -jar hudson.war --httpPort=8000
```



Saturday, December 17, 11



# 应用持续集成时常遇到的问题

- 频繁构建习惯(合适的轮询规则设定)
- Ant脚本与程序同等重要
- 合理规划版本库布局

# 推荐阅读书籍与站点

- 官方文档  
(flexantTasks/flexunitTasks/compile args)
- Martin Fowler相关文章  
<http://martinfowler.com/articles/continuousIntegration.html>
- 书籍  
《持续交付-发布可靠软件的系统方法》

谢谢 ~ ~

<http://weibo.com/agilelife/>

<https://github.com/jexchan/9riapreso>