AgileApps Cloud

The Fast Path to
Agile Process-Driven Applications

Get There Faster

# Development Techniques

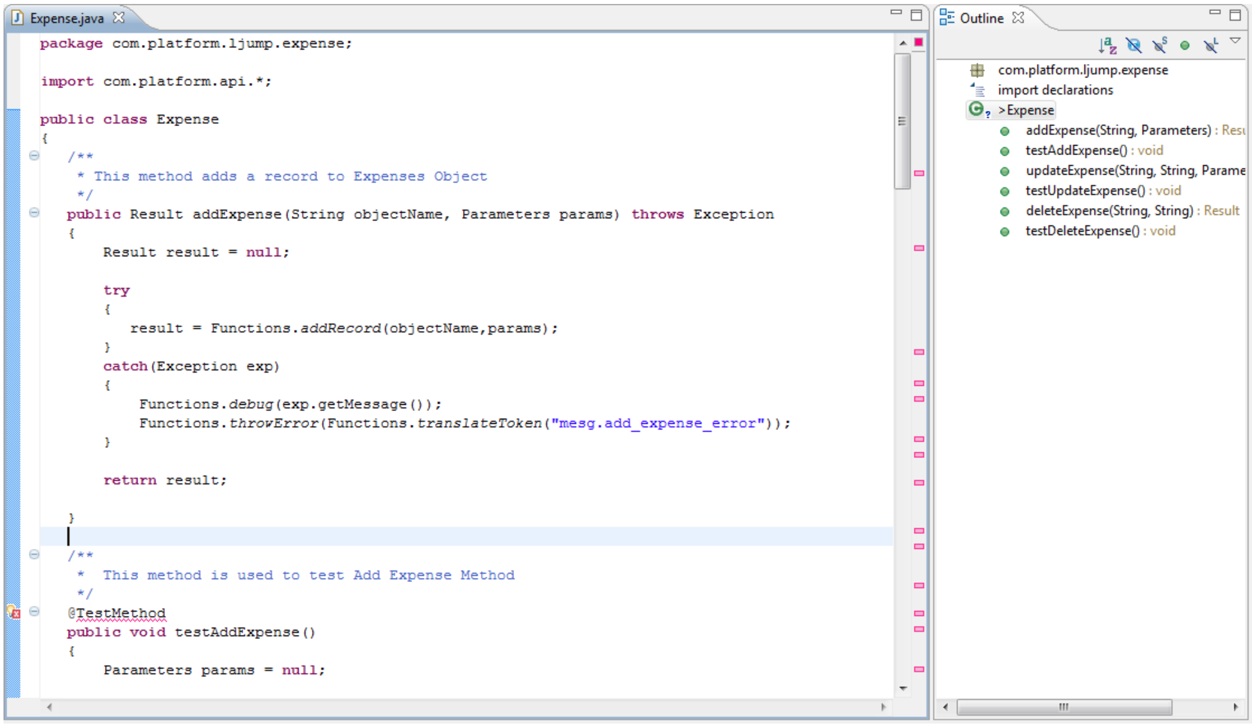
## in the AgileApps Cloud

## Contents

# 1. IDEs and Unit Testing

## 1.1 Developing Classes and Pages with the Eclipse Plug-in

Interactive Development Environments (IDEs) make development easier. Developers can use the Eclipse plug-in to develop Classes and Pages:


Expense.java
Outline
com.platform.ljump.expense
import declarations
>Expense
addExpense(String, Parameters) : Resu
testAddExpense() : void
updateExpense(String, String, Parame
testUpdateExpense() : void
deleteExpense(String, String) : Result
testDeleteExpense() : void

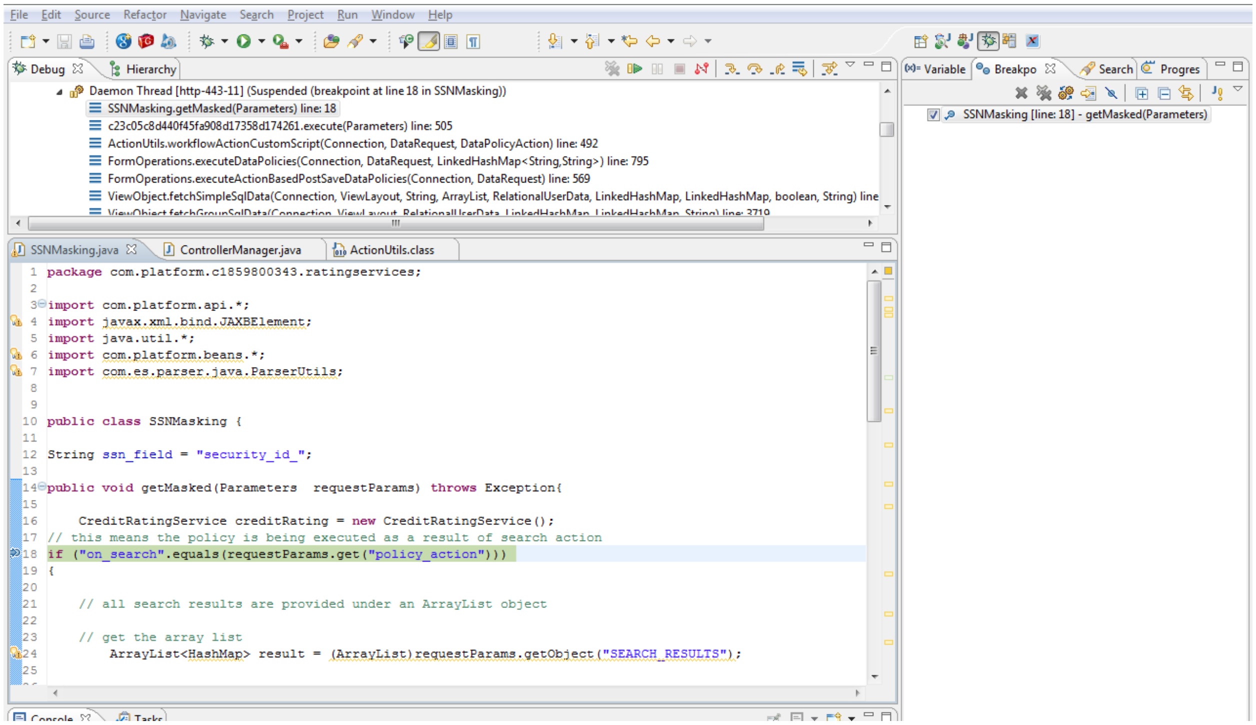
```
package com.platform.ljump.expense;

import com.platform.api.*;

public class Expense
{
    /**
     * This method adds a record to Expenses Object
     */
    public Result addExpense(String objectName, Parameters params) throws Exception
    {
        Result result = null;

        try
        {
           result = Functions.addRecord(objectName,params);
        }
        catch(Exception exp)
        {
            Functions.debug(exp.getMessage());
            Functions.throwError(Functions.translateToken("mesg.add_expense_error"));
        }

        return result;

    }

    /**
     *  This method is used to test Add Expense Method
     */
    @TestMethod
    public void testAddExpense()
    {
        Parameters params = null;
```

*Learn more:* http://agileappslive.info/wiki/Eclipse_Plug-In

## 1.2 Debugging with the Eclipse Plug-in

AgileApps allows classes and pages to be debugged using Eclipse Plugin in development mode. Standard debugging configuration in Eclipse is used for this purpose.

The project should be configured to build automatically.



```
Daemon Thread [http-443-11] (Suspended (breakpoint at line 18 in SSNMasking))
    SSNMasking.getMasked(Parameters) line: 18
    c23c05c8d440f45fa908d17358d174261.execute(Parameters) line: 505
    ActionUtils.workflowActionCustomScript(Connection, DataRequest, DataPolicyAction) line: 492
    FormOperations.executeDataPolicies(Connection, DataRequest, LinkedHashMap<String,String>) line: 795
    FormOperations.executeActionBasedPostSaveDataPolicies(Connection, DataRequest) line: 569
    ViewObject.fetchSimpleSqlData(Connection, ViewLayout, String, ArrayList, RelationalUserData, LinkedHashMap, LinkedHashMap, boolean, String) line
    ViewObject.fetchGroupSqlData(Connection, ViewLayout, RelationalUserData, LinkedHashMap, LinkedHashMap, String) line: 3719
```

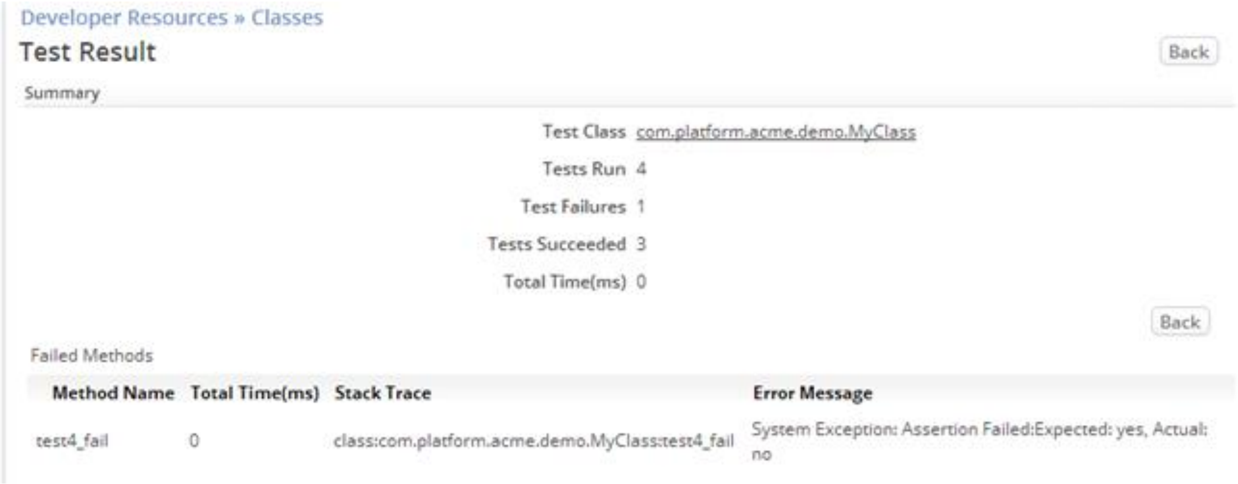
```
SSNMasking [line: 18] - getMasked(Parameters)
```

```java
package com.platform.c1859800343.ratingservices;

import com.platform.api.*;
import javax.xml.bind.JAXBElement;
import java.util.*;
import com.platform.beans.*;
import com.es.parser.java.ParserUtils;


public class SSNMasking {

String ssn_field = "security_id_";

public void getMasked(Parameters  requestParams) throws Exception{

    CreditRatingService creditRating = new CreditRatingService();
// this means the policy is being executed as a result of search action
if ("on_search".equals(requestParams.get("policy_action")))
{

    // all search results are provided under an ArrayList object

    // get the array list
        ArrayList<HashMap> result = (ArrayList)requestParams.getObject("SEARCH_RESULTS");

```

**Note:**
Sometimes the code optimization done by Eclipse is different than the optimization done by the JVM in tomcat. Source code linking issues can result. To work around that problem, make a small change to the source code while debugging (for example, by adding/removing a space) and save. The Eclipse debugger will immediately link to the source code correctly, allowing developers to step through the code as well as hot-deploy any changes.

## 1.3 Unit Testing

Developers can use the integrated unit testing environment in AgileApps to create and launch unit tests. Here are the results from running the tests defined in a single class:



Test Result

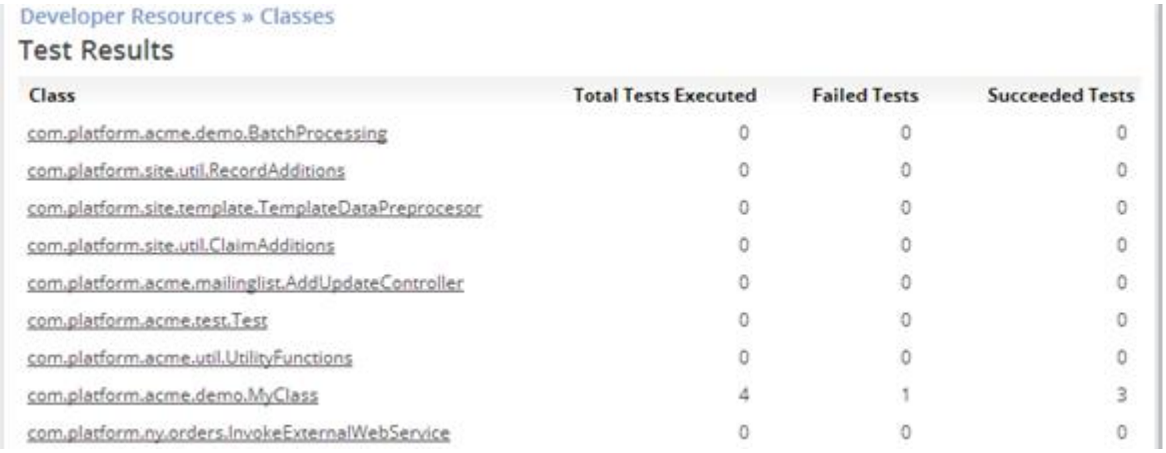
Summary

| | |
|---|---|
| Test Class | com.platform.acme.demo.MyClass |
| Tests Run | 4 |
| Test Failures | 1 |
| Tests Succeeded | 3 |
| Total Time(ms) | 0 |

Failed Methods

| Method Name | Total Time(ms) | Stack Trace | Error Message |
|---|---|---|---|
| test4_fail | 0 | class:com.platform.acme.demo.MyClass:test4_fail | System Exception: Assertion Failed:Expected: yes, Actual: no |

Tests on all classes can be run at one time, as well:



Test Results

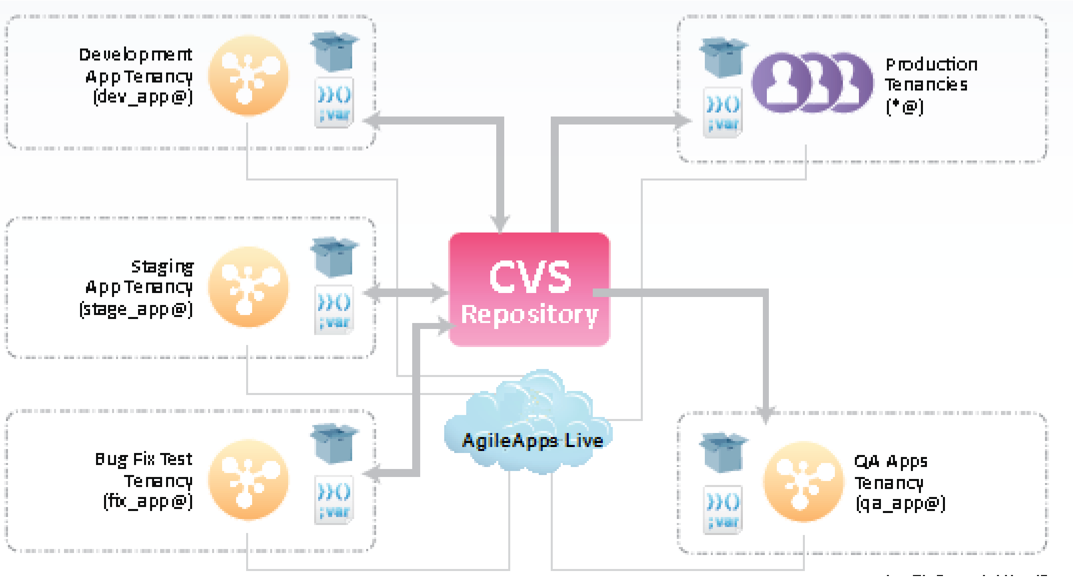
| Class | Total Tests Executed | Failed Tests | Succeeded Tests |
|---|---|---|---|
| com.platform.acme.demo.BatchProcessing | 0 | 0 | 0 |
| com.platform.site.util.RecordAdditions | 0 | 0 | 0 |
| com.platform.site.template.TemplateDataPreprocesor | 0 | 0 | 0 |
| com.platform.site.util.ClaimAdditions | 0 | 0 | 0 |
| com.platform.acme.mailinglist.AddUpdateController | 0 | 0 | 0 |
| com.platform.acme.test.Test | 0 | 0 | 0 |
| com.platform.acme.util.UtilityFunctions | 0 | 0 | 0 |
| com.platform.acme.demo.MyClass | 4 | 1 | 3 |
| com.platform.ny.orders.InvokeExternalWebService | 0 | 0 | 0 |

*Learn more:* http://agileappslive.info/wiki/Unit_Test_Framework

# 2. Development Models

## 2.1 Cloud Development Model

In the Cloud Development model all the developers are located on the same tenancy in AgileApps. They all have User IDs in a shared development environment.



After developers have created the first version of their application, they create a package containing the application and publish it to the QA tenancy. After the QA has certified the package it can be promoted to staging and production tenancies or published into the catalog. Developers also have the option to download the package ZIP file and check it into their organization's version control repository.

## 2.2 Sandbox Development Model

In the sandbox model, developers and testers have their own tenancy in which to carry out their operations. Packages are published following defined paths, so that developers can share developments with each other, for example, and so they can publish to a common Q/A tenancy—but only the Q/A tenancy is allowed to publish an application package to the production tenancy.

*Learn more:* http://agileappslive.info/wiki/Sandboxes

# 3. Deployment

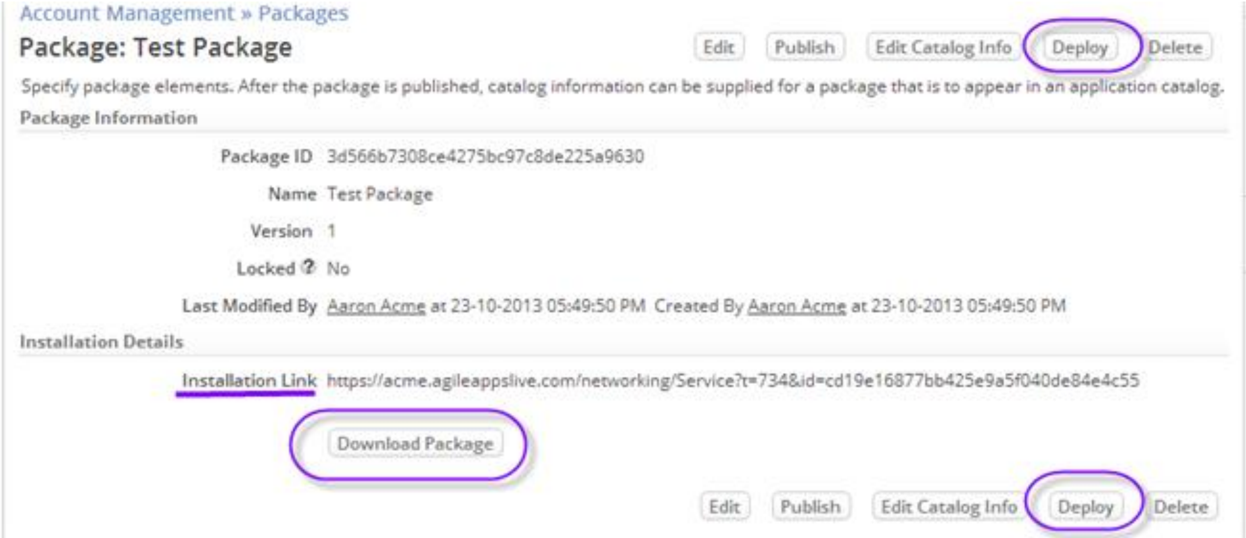
## 3.1 Publishing to the Community Marketplace

After a fully tested package has been loaded into the production tenancy, the developer has an option to use the AgileApps application-listing management system to create a listing request. As part of the request, the developer can provide information for the listing. Once the request has been approved, the application appears in the market place.

*Learn more:* [http://agileappslive.info/wiki/Community_Marketplace](http://agileappslive.info/wiki/Community_Marketplace)

## 3.2 Deploying Packages to Tenants and Others

Packages have multiple deployment options. As shown here, a package zip-file can be downloaded directly, the link can be copied and placed in a location for others to find, or tenants who have installed the package can be upgraded directly, by choosing the Deploy option.



Account Management » Packages

Package: Test Package

Specify package elements. After the package is published, catalog information can be supplied for a package that is to appear in an application catalog.

Package Information

Package ID 3d566b7308ce4275bc97c8de225a9630

Name Test Package

Version 1

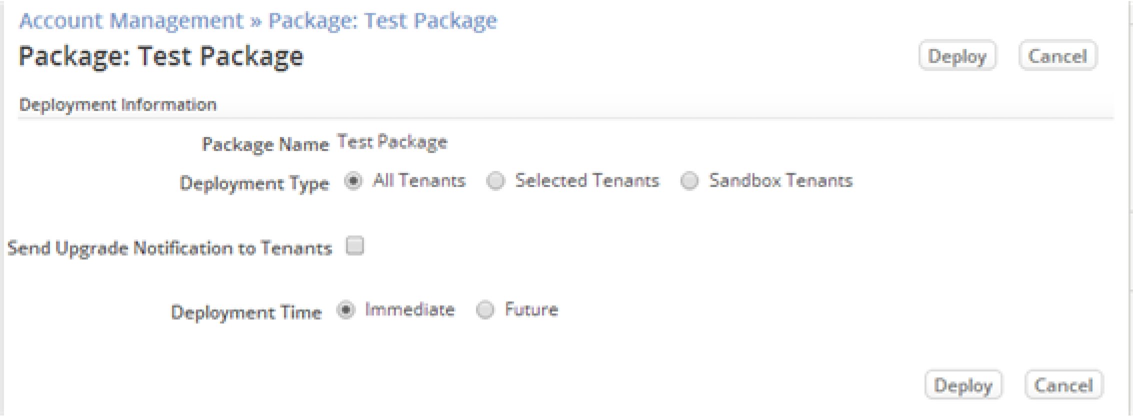
Locked ? No

Last Modified By Aaron Acme at 23-10-2013 05:49:50 PM Created By Aaron Acme at 23-10-2013 05:49:50 PM

Installation Details

Installation Link https://acme.agileappslive.com/networking/Service?t=734&id=cd19e16877bb425e9a5f040de84e4c55

Packages can be directly deployed to all tenants who have subscribed to the package, to selected tenant subscribers, or to selected sandbox tenants, either immediately or at a scheduled time in the future:



Account Management » Package: Test Package

Package: Test Package

Deployment Information

Package Name Test Package

Deployment Type ● All Tenants ○ Selected Tenants ○ Sandbox Tenants

Send Upgrade Notification to Tenants □

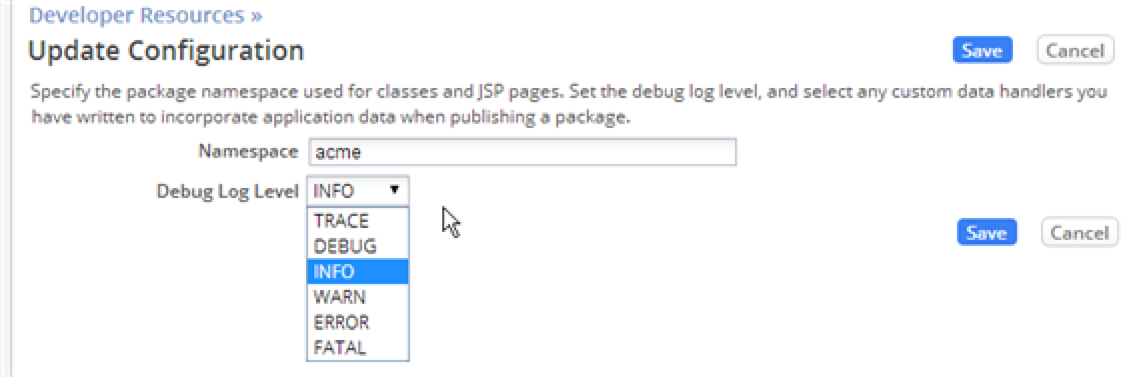
Deployment Time ● Immediate ○ Future

REST APIs can also be used to submit the deployment job. The API Call returns a Job ID, which can used to check the status of the job. At the end of the deployment process, the system sends summary deployment information to the job submitter.

*Learn more:* [http://agileappslive.info/wiki/Packages#Deploy_a_Package](http://agileappslive.info/wiki/Packages#Deploy_a_Package)

## 4. Tracing Issues in a Production Environment

Once Applications have been deployed into customer tenancies, it is possible for developers to "proxy-login" to those tenancies using the built-in Customer Support system. Once there, developers can enable a variety of debug-log levels.



Developer Resources »

Update Configuration

Specify the package namespace used for classes and JSP pages. Set the debug log level, and select any custom data handlers you have written to incorporate application data when publishing a package.

Namespace: acme

Debug Log Level: INFO

Learn more:

- [http://agileappslive.info/wiki/Proxy_Login_as_this_User](http://agileappslive.info/wiki/Proxy_Login_as_this_User)
- [http://agileappslive.info/wiki/Developer_Configuration](http://agileappslive.info/wiki/Developer_Configuration)

## 5. Change Control

AgileApps can be configured to allow application change rights for authorized system roles. For example, some roles might be able to add Java classes, some may only be able to add or customize layouts.