Hands-On Lab

# Collaboration Experiences for Development Teams using Team Foundation Server 2015

Lab version: 14.0.23107.0

Last updated: 9/11/2015

* 1. 

1. **TABLE OF CONTENT**

[Collaboration Experiences for Development Teams using Team Foundation Server 2015 1](#_Toc430678325)

[Overview 3](#_Toc430678326)

[Prerequisites 3](#_Toc430678327)

[About the Fabrikam Fiber Scenario 3](#_Toc430678328)

[Exercises 3](#_Toc430678329)

[Exercise 1: Team Room Collaboration 4](#_Toc430678330)

[Task 1: Team Rooms Introduction 4](#_Toc430678331)

[Task 2: Team Room Mentions and Links 6](#_Toc430678332)

[Task 3: Team Room Events 9](#_Toc430678333)

[Exercise 2: Lightweight Code Comments 14](#_Toc430678334)

[Exercise 3: CodeLens 21](#_Toc430678335)

## Overview

* 1. In this lab, you’ll learn about some of the features in Visual Studio 2015 and Team Foundation Server 2015 that enable collaboration experiences for development teams including Team Rooms, Lightweight Code Comments, and CodeLens.

### Prerequisites

* 1. In order to complete this lab you will need the Visual Studio 2015 virtual machine provided by Microsoft. For more information on acquiring and using this virtual machine, please see [this blog post](http://aka.ms/ALMVM).

### About the Fabrikam Fiber Scenario

* 1. This set of hands-on-labs uses a fictional company, Fabrikam Fiber, as a backdrop to the scenarios you are learning about. Fabrikam Fiber provides cable television and related services to the United States. They are growing rapidly and have embraced Windows Azure to scale their customer-facing web site directly to end-users to allow them to self-service tickets and track technicians. They also use an on-premises ASP.NET MVC application for their customer service representatives to administer customer orders.

In this set of hands-on labs, you will take part in a number of scenarios that involve the development and testing team at Fabrikam Fiber. The team, which consists of 8-10 people, has decided to use Visual Studio application lifecycle management tools to manage their source code, run their builds, test their web sites, and plan and track the project.

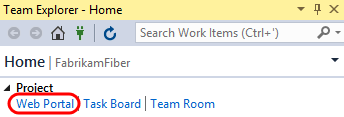
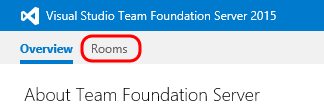
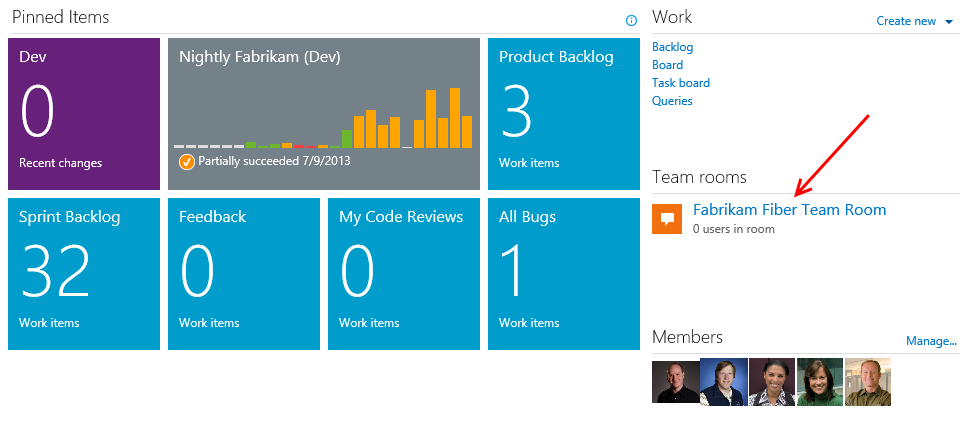
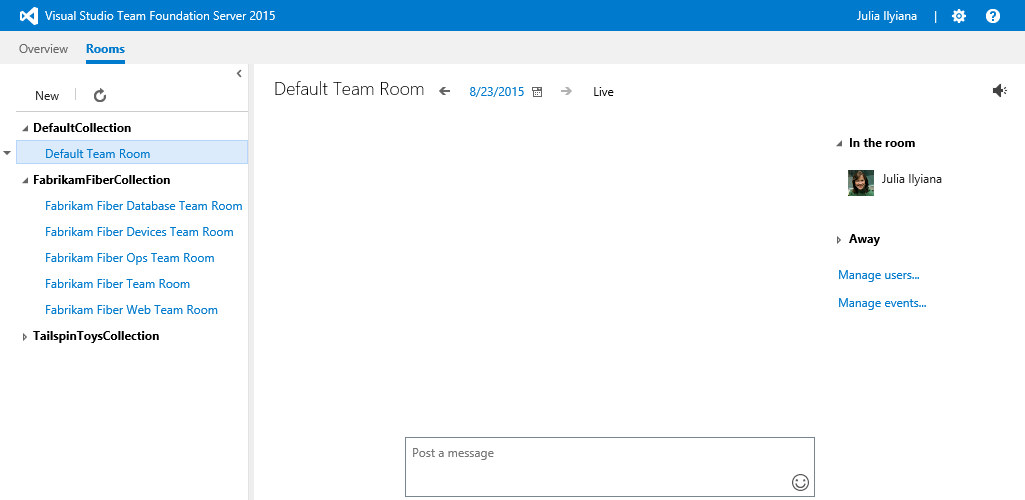
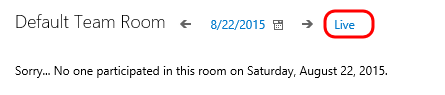
### Exercises

* 1. This hands-on lab includes the following exercises:
  2. Team Room Collaboration
  3. Lightweight Code Comments
  4. CodeLens
  5. Estimated time to complete this lab: **60 minutes**.

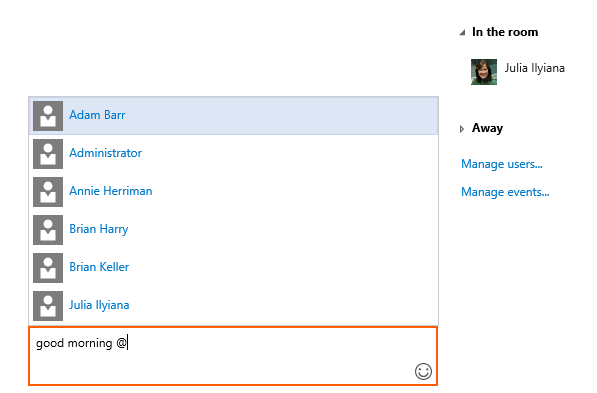
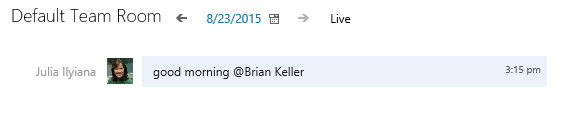
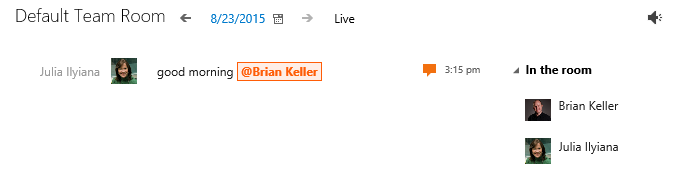
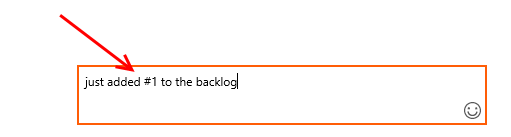
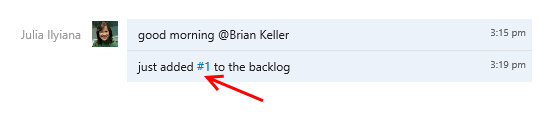
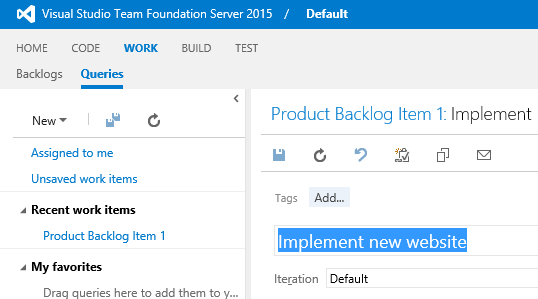
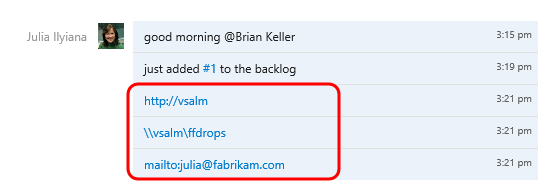
## Exercise 1: Team Room Collaboration

* 1. In this exercise, you will learn about the Team Room feature in Team Foundation Server 2015 (and at visualstudio.com). Team Rooms provide a durable collaboration space where members can chat and view pertinent events, thereby allowing them to remain in loose contact throughout the workday.

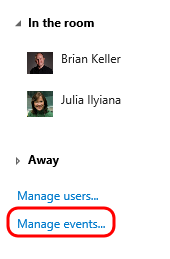
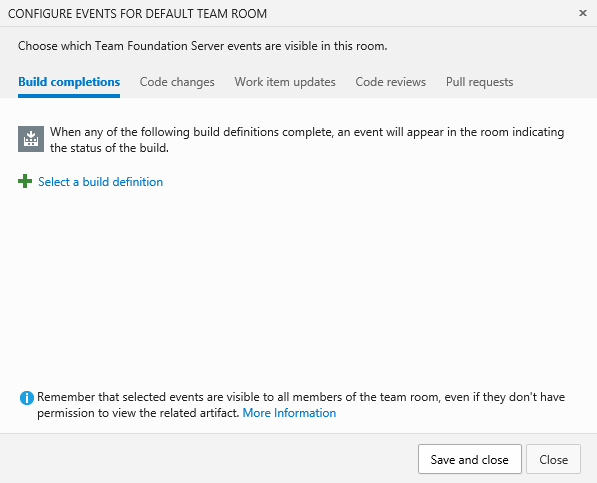
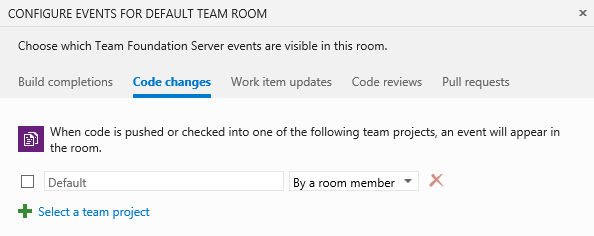
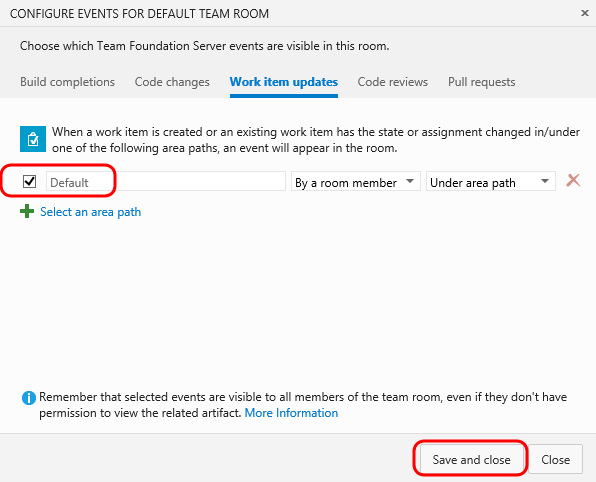
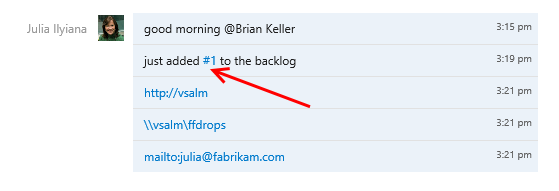
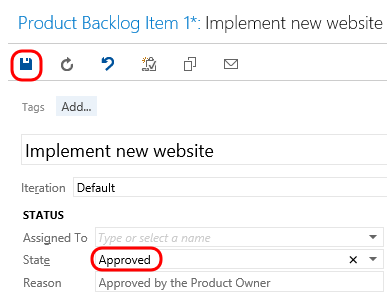
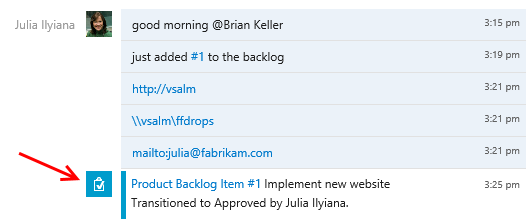
#### Task 1: Team Rooms Introduction

* 1. Log in as **Julia Ilyiana** (VSALM\Julia). All user passwords are **P2ssw0rd**.
  2. Launch **Internet Explorer** from the taskbar and select the **TFS Portal** button from the favorites bar at the top.
     1. 
     2. Figure
     3. Launching the team web access site
     4. **Note:** There is also a **Web Portal** link in the Team Explorer window within Visual Studio.
     5. ****
  3. Select the **Rooms** link to navigate to and enter the **Default Team Room**. If this room does not automatically load, click on the Default Team Room link under the DefaultCollection node.
     1. 
     2. Figure
     3. Navigating to team room
     4. **Note:** There is also a team room tile on the home page of each project.
     5. ****
  4. This team room was automatically created for the project named Default, hence the name “Default Team Room”. On the left side is the **Rooms Explorer** pane, which you can use to navigate between team rooms and perform some administrative capabilities. On the right side is the currently selected team room showing the recorded messages and activity for the current day.
     1. 
     2. Figure
     3. Default team room
  5. Since the team room content is persistent, team members can go back in time to see what happened while they were gone from work. Select the back arrow to the left of the current date.
     1. 
     2. Figure
     3. Selecting previous day in team room
  6. It appears that we didn’t miss anything important in the team room, so we can return to the current date by selecting the **Live** link.
     1. 
     2. Figure
     3. Returning to the live team room

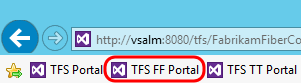
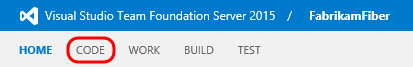
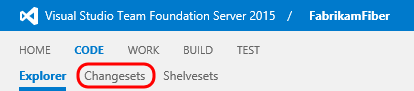
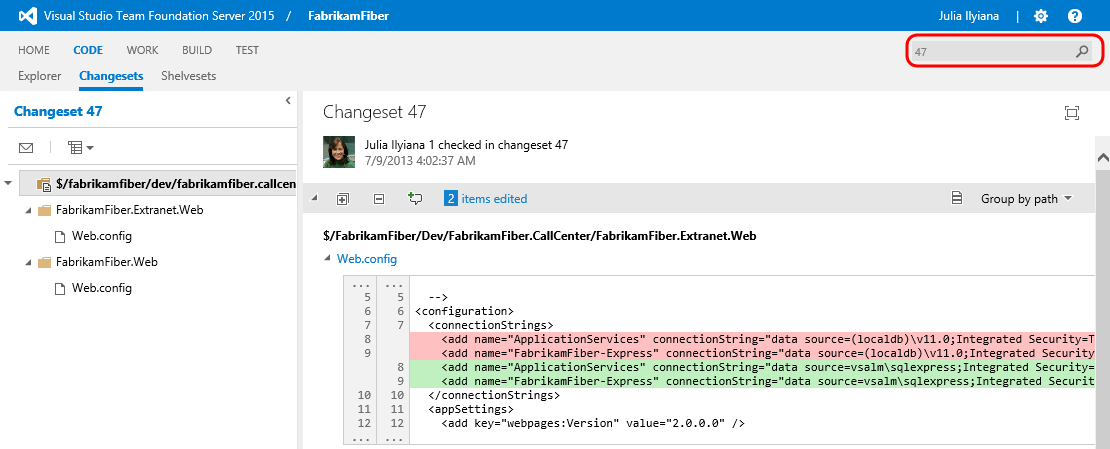
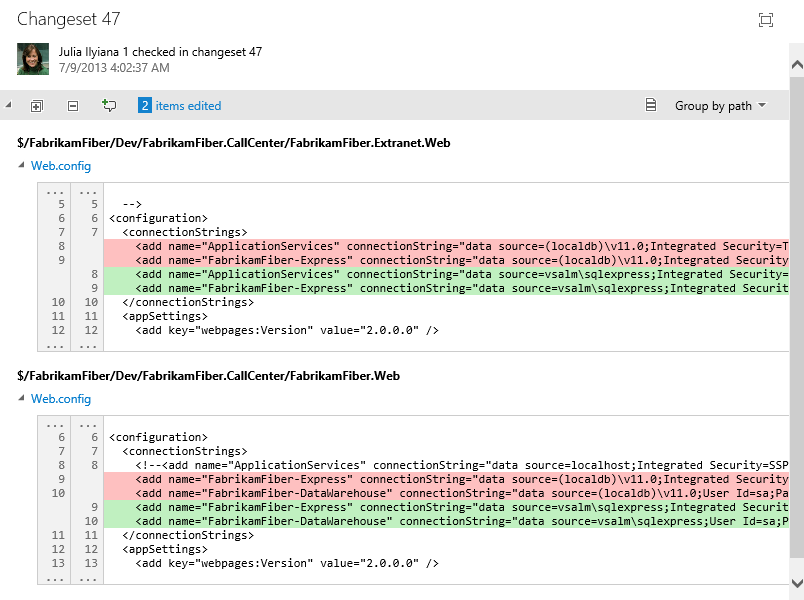
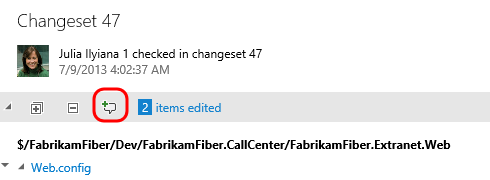
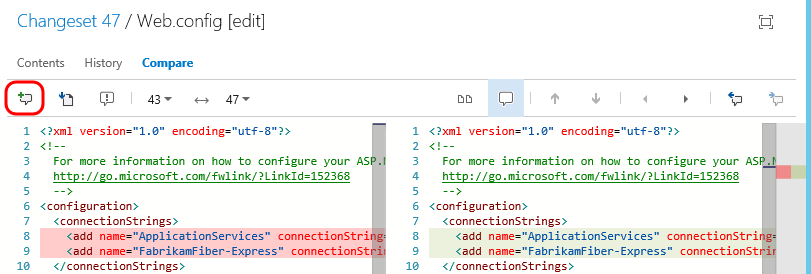
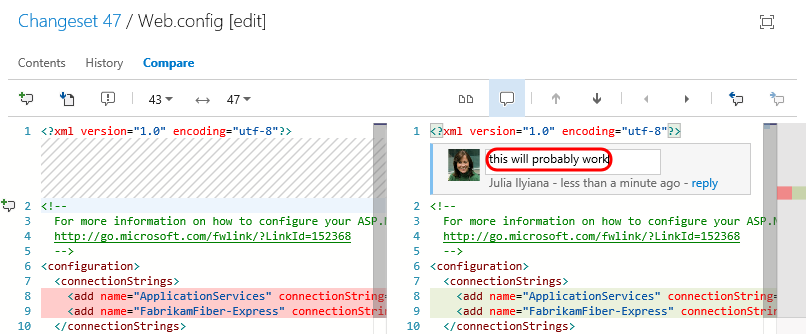
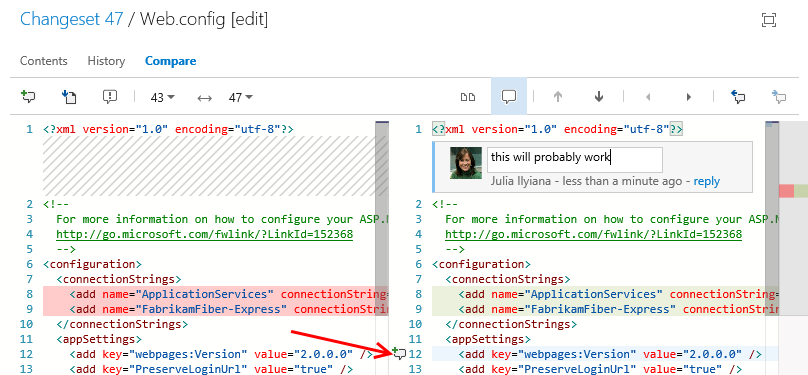
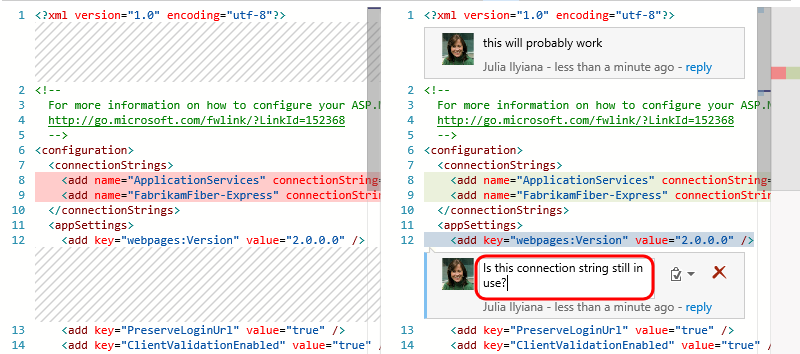
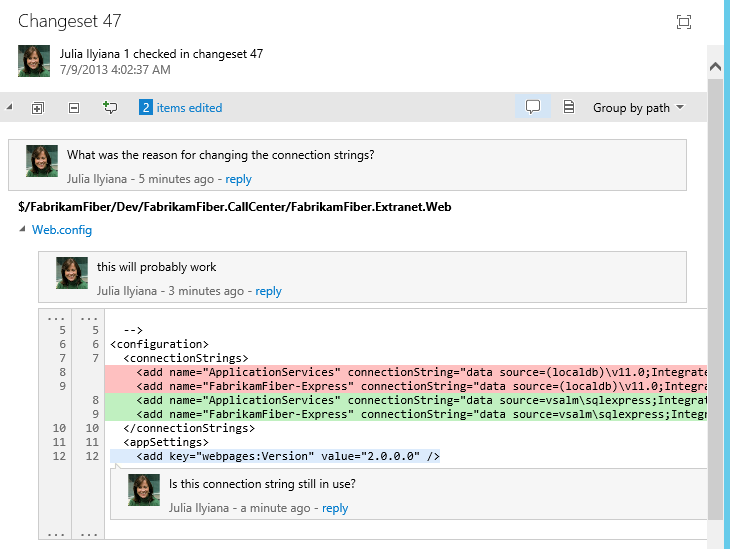
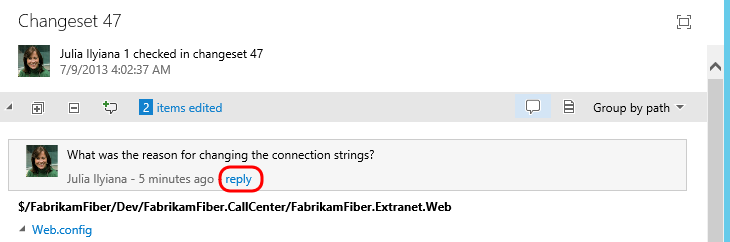
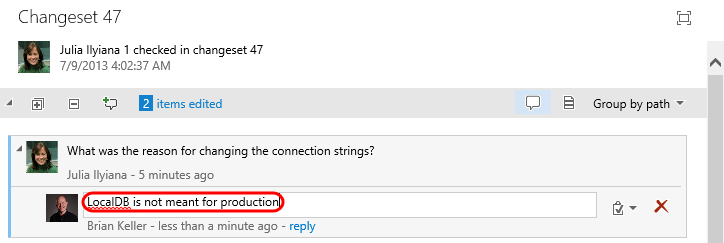
#### Task 2: Team Room Mentions and Links

* 1. Now let’s take a look at some of the types of mentions and links that can be inserted via team room chat. The first type is the team member mention, which can be inserted anywhere by typing the ‘**@**’ symbol followed by the name. As Julia, try this out by typing “**good morning @**” into the textbox at the bottom of the window. Note that a list of room members is presented as soon as you type the ‘@’ symbol.
     1. 
     2. Figure
     3. Creating team member mention
  2. Continue typing **‘Brian Keller**’ or select his name from the list and then press **Enter**.
     1. 
     2. Figure
     3. Team member mention (view from non-target team member)
  3. Before we move on, go ahead and get another team member connected to the room using remote desktop. In the new remote desktop window, login with user name “**VSALM\brian**” and password “**P2ssw0rd**”. Once connected and logged in, navigate to the **Default Team Room** as you previously did.
     1. **Note:** You can also RDP to ‘localhost’ from within the VM itself.
  4. Note that **Brian’s** view of the message from Julia renders differently than it would for others since he was the person addressed.
     1. 
     2. Figure
     3. Team member mention (view from target team member)
  5. Another type of mention can be used for referencing project work items. As **Julia**, add a message to the team that mentions work item #1.
     1. 
     2. Figure
     3. Creating work item mention
  6. Select the work item link from the team room chat to see that it does link to the specified work item.
     1. 
     2. Figure
     3. Hyperlink to work item
     4. 
     5. Figure
     6. Work Item view opens in new browser tab
  7. Close the tab showing the work item to return to the team room chat.
  8. Some other patterns render as links include URLs, UNC network paths, and email addresses.
     1. 
     2. Figure
     3. Other types of links

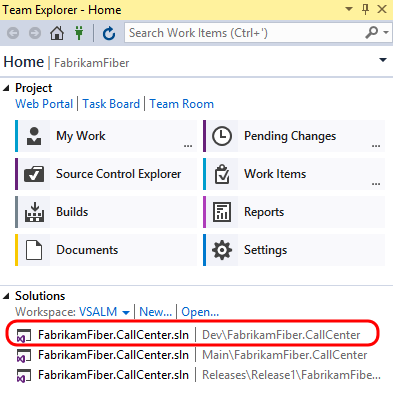
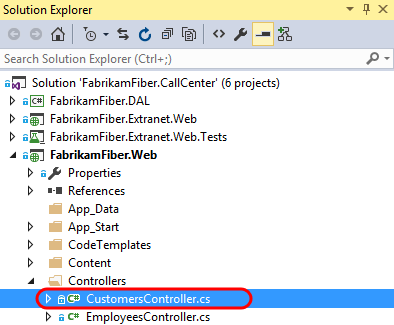
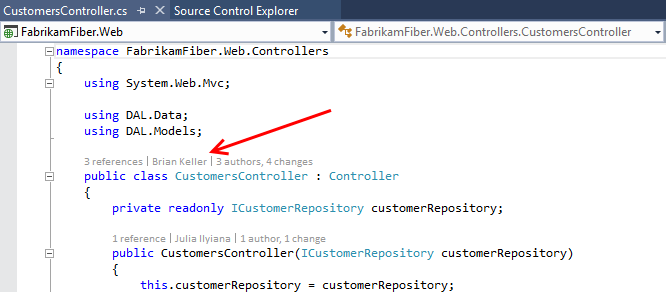
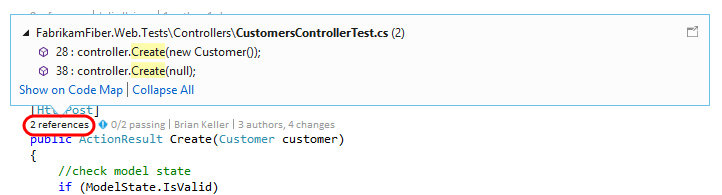
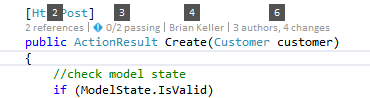
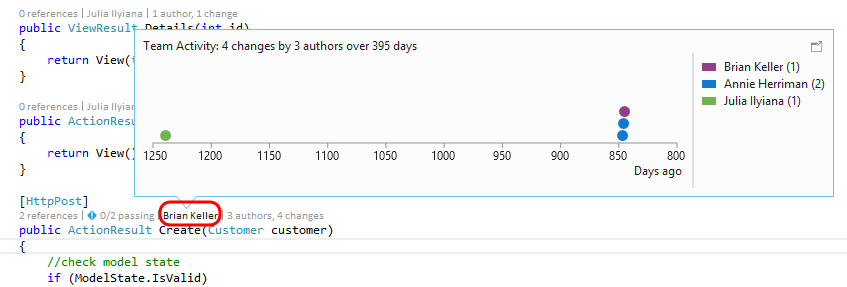
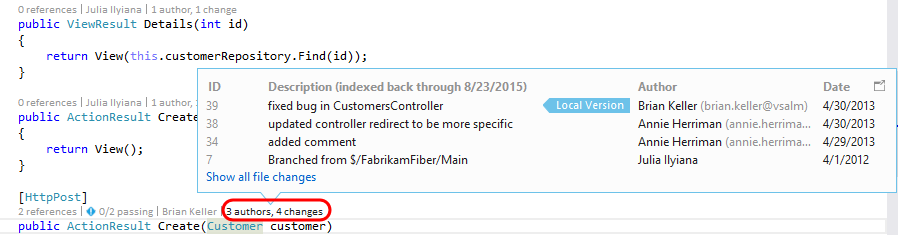
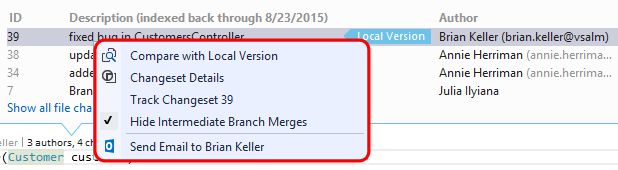
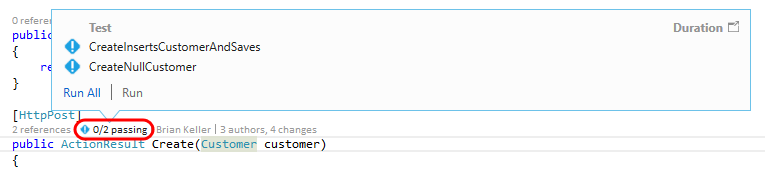
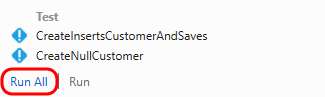
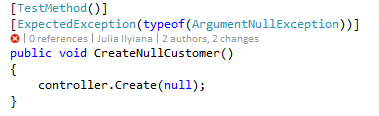
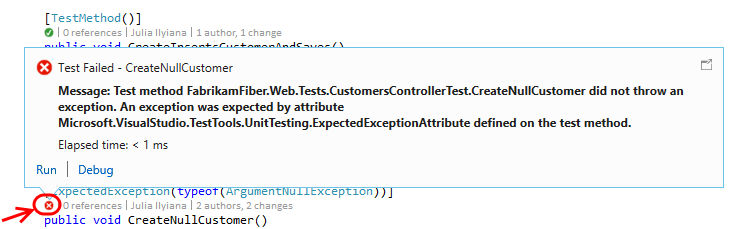
#### Task 3: Team Room Events

* 1. Another key feature of team rooms is that various events can be selected and configured to automatically feed in additional information, thereby providing additional context to active and historical team discussions. Select the “**Manage events…**” link in the Default Team Room.
     1. 
     2. Figure
     3. Manage team room events
  2. The **Configure Events for Default Team Room** window that appears shows four different options, **Build Completion**, Code Changes, Work Item Updates, and Code Reviews. Build completion events can be setup to appear whenever selected builds complete.
     1. 
     2. Figure
     3. Configuring build completion events
  3. Select the **Code Changes** tab. These events can be setup to appear whenever code is checked into selected team projects.
     1. 
     2. Figure
     3. Configuring code change events
  4. Select the **Work Item Updates** tab. These events can be setup to appear whenever a work item is created or an existing work item has the state or assignment changed. Let’s set this event up by checking the checkbox next to the **Default** project and then selecting the **Save and Close** button to use the default options.
     1. 
     2. Figure
     3. Configuring work item update events
  5. Navigate to work item **#1** in TFS web access by selecting the link in the Default Team Room chat window.
     1. 
     2. Figure
     3. Hyperlink to work item
  6. Change the work item from state **New** to **Approved** and then select the **Save** button.
     1. 
     2. Figure
     3. Updating work item state
  7. Return to the Default Team Room and note that an event is raised in the chat window describing the state transition. It should only take a moment for it to show up, but you can also refresh the page after a few minutes if needed.
     1. 
     2. Figure
     3. Work item update notification
  8. The Team Room feature has been designed with extensibility in mind, so that will allow development teams to add in custom events and clients in the future. If you are interested in learning more about this extensibility, look at the [Team Rooms REST API](https://www.visualstudio.com/en-us/integrate/api/chat/overview) for more information.

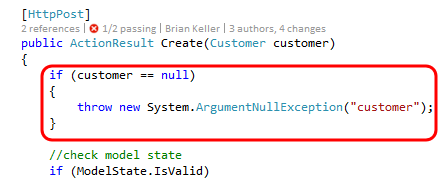
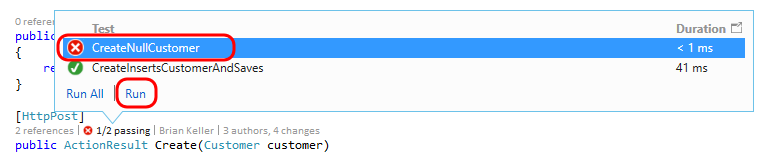
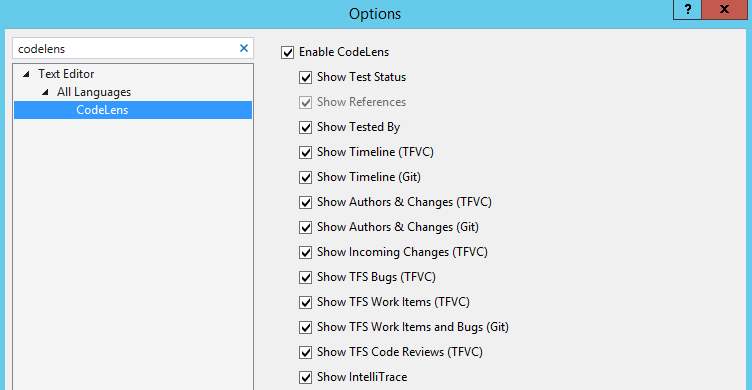
## Exercise 2: Lightweight Code Comments

* 1. In this exercise, you will learn about the Lightweight Code Comment feature (first introduced with Team Foundation Server 2013), that allows team members to comment on code. This enables interactive or time shifted conversations about code, all from a very nice inline experience in the browser.
  2. Log in as **Julia Ilyiana** (VSALM\Julia). All user passwords are **P2ssw0rd**.
  3. Launch **Internet Explorer** from the taskbar and select the **TFS FF Portal** button from the favorites bar at the top.
     1. 
     2. Figure
     3. Launching web access
  4. Select the **Code** link.
     1. 
     2. Figure
     3. Navigating to Code
  5. Select the **Changesets** link. Lightweight code comments can be applied to both changesets and shelvesets.
     1. 
     2. Figure
     3. Navigating to Changesets
  6. Load changeset 47 by entering “**47**” into the changeset search box and pressing **Enter**.
     1. 
     2. Figure
     3. Changeset link
  7. This changeset contains two file edits, with the changes highlighted in the diff viewer.
     1. 
     2. Figure
     3. Diff viewer showing file edits
  8. Select the **Add Comment** button to add a comment to the changeset.
     1. 
     2. Figure
     3. Comment button location
  9. Enter a comment such as “**What was the reason for changing the connection strings?**” Press **Enter** when done.
     1. 
     2. Figure
     3. Adding comment to changeset
  10. To comment on a specific file, select the first **Web.config** link to load that specific edit.
      1. 
      2. Figure
      3. Loading diff viewer for a specific edit
  11. Select the **Add Comment** button.
      1. 
      2. Figure
      3. Add comment button location
  12. Add a short comment of your choosing.
      1. Figure
      2. Comment added to a specific file edit
  13. Move the mouse cursor over the lines of the file and note that the same **Add Comment** button appears in the margin. Go ahead and select that button while holding the mouse cursor over one of the lines to add a line comment of your choice.
      1. 
      2. Figure
      3. Adding a line comment
      4. 
      5. Figure
      6. Example result of adding line comment
      7. **Note:** You can even highlight parts of a sentence and comment on them if desired.
  14. Connect to the VM as **Brian** using the same method that you chose in exercise 1, and then navigate to **changeset 47** in the Fabrikam Fiber web portal. Note that you can see the comments left by Julia.
      1. 
      2. Figure
      3. Viewing comments left by Julia (as Brian)
  15. Select the **Reply** link within Julia’s first comment and leave a response to her question.
      1. 
      2. Figure
      3. Replying to Julia (as Brian)
      4. 
      5. Figure
      6. Result of replying to Julia (as Brian)

## Exercise 3: CodeLens

* 1. In this exercise, you will learn about the CodeLens feature first introduced with Visual Studio 2013 and Team Foundation Server 2013 that provides code insights about classes, methods, and properties directly within the code editor.
  2. Log in as **Julia Ilyiana** (VSALM\Julia). All user passwords are **P2ssw0rd**.
  3. Launch **Visual Studio 2015** from the taskbar.
  4. In **Team Explorer – Home**, double-click on the first solution listed, “**FabrikamFiber.CallCenter.sln**”.
     1. 
     2. Figure
     3. Loading the Fabrikam Fiber Call Center solution
  5. In **Solution Explorer**, expand the **FabrikamFiber.Web | Controllers** folder and double-click on **CustomersController.cs** to open it.
     1. 
     2. Figure
     3. Opening CustomersController.cs
  6. After loading the CustomersController.cs file in the code editor, note that both the class and its methods show a single line of indicators. These indicators are meant to provide developers a heads-up-display of sorts - with information about how the code is being used, tested, and changed.
     1. **Note:** CodeLens is available for managed code only.
     2. 
     3. Figure
     4. CodeLens indicators
  7. Scroll down to locate the **Create** method that takes a Customer object as a parameter, and then select the **references indicator**. This shows other code references to this method.
     1. 
     2. Figure
     3. References indicator
     4. **Note:** You can also hold down the **Alt** key and press the number key shown above each indicator as a shortcut.
     5. ****
  8. If you were to double-click on one of the references, you would be taken to the specified location in code. Press the **Escape** key instead.
  9. Select the **Timeline** indicator above the same Create method as before. Note that before selecting it, you can see who made the most recent change.
     1. 
     2. Figure
     3. Author indicator
  10. Select the **Authors & Changes** indicator to view all authors, changeset descriptions, and dates.
      1. 
      2. Figure
      3. Author code lens
  11. **Right-click** on one of the rows shown for the expanded **Changes** indicator. Note that you can choose to view the diff, view the changeset details, track the changset, get this version, or even send email to the author.
      1. 
      2. Figure
      3. Additional options for indicator
  12. Press the **Escape** key.
  13. Press **Ctrl + Shift + B** to build the solution.
  14. Select the **Tested By** indicator shown above the Create method. This lens currently shows that associated tests have not been executed.
      1. 
      2. Figure
      3. Tested By indicator
  15. Select the **Run All** link to execute the tests.
      1. 
      2. Figure
      3. Running all tests associated with the Create method
  16. It appears that one of the tests failed, so select the **Tested By** indicator once again to see which test is causing a problem.
      1. 
      2. Figure
      3. Tested By indicator showing failed test
  17. Let’s take a look at the test itself to determine if the fix needs to happen there or in the Create method being tested. **Double-click** on the **CreateNullCustomer** test to navigate to the definition.
      1. 
      2. Figure
      3. Navigating to test source
  18. After looking at the **CreateNullCustomer** test, we can see that it is expecting an **ArgumentNullException** to be thrown when the Create method is called with a null parameter. Note that there is also a **Test Status** indicator shown next to the test method itself.
      1. 
      2. Figure
      3. Test method showing most recent test result indicator
  19. Select the indicator showing the failed **CreateNullCustomer** test to see additional summary information.
      1. 
      2. Figure
      3. Viewing test result summary
  20. Throwing this exception seems like a reasonable expectation, so let’s go ahead and make the fix in the Create method. Return to the **CustomersController.cs** file and add the following code to the beginning of the Create method that takes a Customer parameter.
      1. C#

if (customer == null)

* + 1. { throw new System.ArgumentNullException("customer"); }
    2. 
    3. Figure
    4. Fixing the Create method
  1. With the fix in place, select the **Tested By** indicator, single-click on the **CreateNullCustomer** test, and then select the **Run** link to re-run just the failed test.
     1. 
     2. Figure
     3. Re-run tests
  2. All tests that reference the Create method now pass.
     1. 
     2. Figure
     3. All tests passing
     4. **Note:** With all tests passing, nothing stands out visually in the CodeLens indicator line. This demonstrates one of the goals of CodeLens - to provide useful information without getting in the way or being distracting. Nevertheless, the feature can be customized or even turned off in **Tools | Options** if desired.
     5. ****
  3. As a final note, CodeLens also supports a number of indicators for Git repositories also. This provides author, change and work item indicators that work very similarly to their Team Foundation Version Control counterparts. Unlike the CodeLens support for TFVC, CodeLens for Git is computed solely on the client side. This means that you can use the feature for any Git repository, whether it is local, cloned from TFS Git, for even cloned from GitHub or another Git source.

To give feedback please write to [VSKitFdbk@Microsoft.com](mailto:VSKitFdbk@Microsoft.com)

Copyright © 2015 by Microsoft Corporation. All rights reserved.