Portfolio 5 (A4154 Name: Ooi Ching Yean(Kris))

The report is intended to:
1. explain the need for Version Control software (such as Git) to organize and manage team-based software development and
2. Show the usage of git on one of the warren repositories.

Version control merging (Team Foundation Server)

Check In Pending Changes is done through Visual Studio 2010.

When you check in files, the system commits pending changes in your workspace to the Team Foundation version control server. This saves any changes made to the files.

Required Permissions
To perform these procedures, you must have the Check in permission set to Allow.

<u>Displaying and checking in pending changes:</u>
To display and check in pending changes

1. View pending changes by performing any one of the following steps:
    - In Source Control Explorer, right-click any file that contains pending changes (indicated by this symbol: ✓), and then click Check In Pending Changes.
    - In Solution Explorer, right-click one or more solutions, projects, or files, and then click Check In.
    - At the Visual Studio Command Prompt, type cd *Path*, where *Path* specifies the directory that contains your updated files, and press ENTER. Then type tf checkin, and press ENTER.
    - On the View menu, point to Other Windows, and click Pending Changes.
2. In the Source Files channel, clear the check boxes for any files or other items that you do not want to check in, and type any applicable comments in the Comment box.
3. Click Check In.
4. The items are checked in, or the Conflicts channel appears with the list of conflicts that you must resolve.

Check out and edit files

When you want to edit a file you can open it from Solution Explorer or from Source Control Explorer. When you begin editing a file, it is automatically checked out to you.

Required permissions:
You must be one of the Contributors for your team project.

To manually check out items:

1. In Solution Explorer select the files that you want to edit, open their shortcut menu, and choose Check Out for Edit.
2. Choose Check Out.

In Solution Explorer and in Source Control Explorer, a check mark ✓ appears next to the items that you have checked out. In Team Explorer, a pending edit change to the file is displayed on the Pending Changes page. In Source Control Explorer, a pending edit change is visible to you and to members of your team.

Merge and resolve conflicts when checking files in

Conflicts can occur when you check files in under the following conditions:

- The date stamp on your file is older than the version that is checked in to the Team Foundation server. This can occur if you get an older version of the file and then try to check it in.
- Other team members have checked in changes between the time that you got the latest version of the file and the time that you try to check it in. In this case, there are two kinds of changes that can occur, even in the same file:
  - If you changed different lines in the file than the other team members, these are considered simple changes and can be merged automatically using the Auto Merge All feature.
  - If you changed the same lines in the file as other team members, these are considered conflicts. Conflicts are more complex and require manual resolution.

Resolving conflicts can seem complex at first, but it can be helpful to understand the different kinds of file conflicts and the different kinds of resolutions

When conflicts occur during a check-in, you initially have two choices:

- **Auto Merge All** . Merges the conflicts automatically. The merge options used by the automatic merging feature should already have been configured by the Microsoft Team Foundation server administrator in Visual Studio 2010 Team System. The automatic merging feature is available only if you made changes to different lines of the file than other team members did, because in that situation, merging is a simple process. Otherwise, you must resolve conflicts manually.

- **Resolve**. Resolves the conflicts manually. The Resolve Version Conflict dialog box will appear, in which you will be given the following additional options:

  1. **Merge changes for me** Uses the automatic merging feature. This option is disabled if Expression Blend detects that you have changed the same lines in the file as another team member.

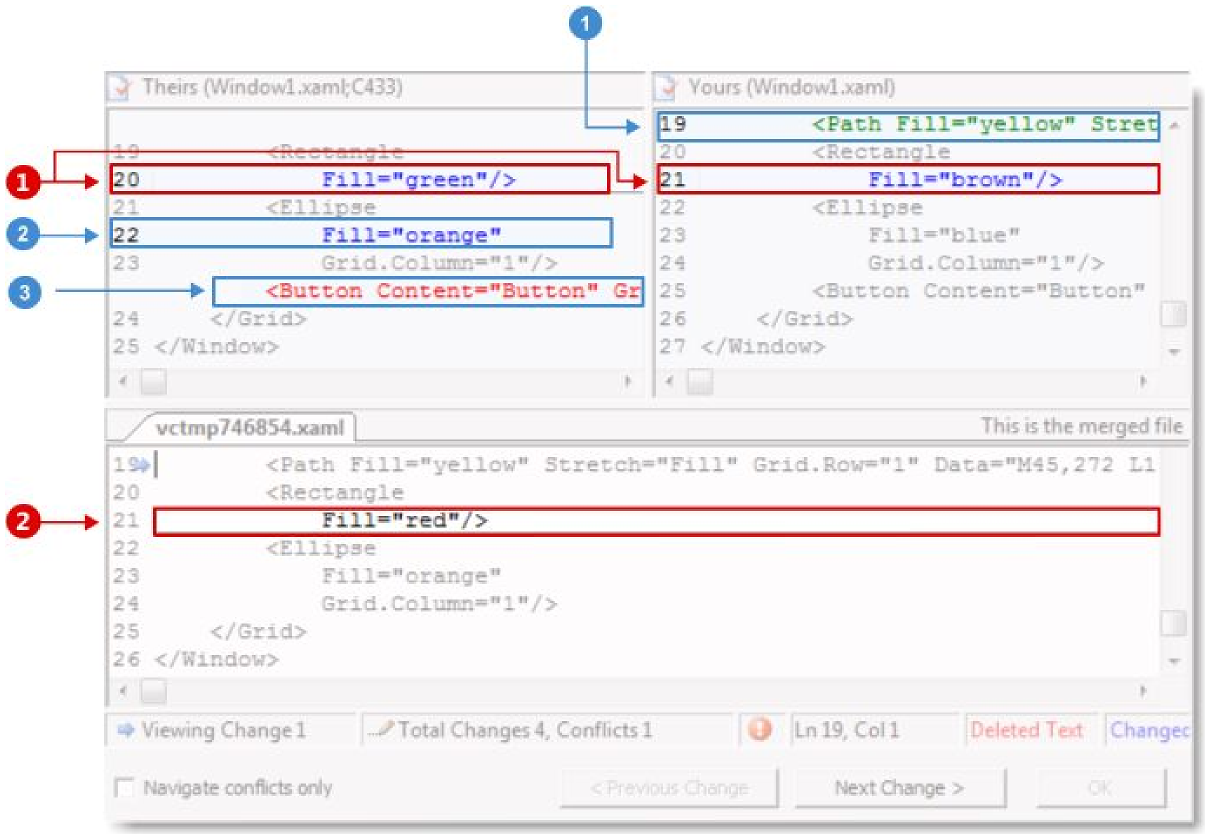
  2. **Compare** Displays a window that compares the differences between any two of the following three versions of the file: your local copy, the copy that was most recently checked by another team member while you had the file checked out, and the original that existed before either one of you checked the file out. Comparing versions is useful in helping you decide whether you want to undo your local changes, discard your server changes, or merge and resolve conflicts using the merge tool.

Using the merge tool

The merge tool gives you the most control because you can select individual conflicting lines and apply them to or remove them from the merged file.

At the top of the merge tool, two panes display different versions of the file. The version that was checked in by the other team member while you had the file checked out is displayed on the left, and your local copy (not yet checked in) is displayed on the right. In each pane, colors are used to identify the lines that have been changed from the original version of the file that existed before the other team member checked in his or her changes. Red lines represent text that was deleted, green lines represent text that was inserted, and blue lines represent text that was modified. If there are changes in the same line in both files in the top two panes, this change is considered a conflict.

At the bottom of the merge tool is a third pane. Initially, this bottom pane displays the original version of the file to which the non-conflicting changes from the top two files have been applied.



```
        <Rectangle
            Fill="green"/>
        <Ellipse
            Fill="orange"
            Grid.Column="1"/>
        <Button Content="Button" Gr
    </Grid>
</Window>
```

```
        <Path Fill="yellow" Stret
        <Rectangle
            Fill="brown"/>
        <Ellipse
            Fill="blue"
            Grid.Column="1"/>
        <Button Content="Button"
    </Grid>
</Window>
```

```
        <Path Fill="yellow" Stretch="Fill" Grid.Row="1" Data="M45,272 L1
        <Rectangle
            Fill="red"/>
        <Ellipse
            Fill="orange"
            Grid.Column="1"/>
    </Grid>
</Window>
```

🔴 1	A line changed both in your local version and in the currently checked-in version, resulting in a conflict that must be resolved manually.
🔴 2	The original version of the line that is in conflict.
🔵 1	A line that was added by you and is already applied to the merged file below.
🔵 2	A line changed in the most recent check-in that is not in conflict with your changes. This line is already applied to the merged file below and cannot be removed.
🔵 3	A line removed in the currently checked-in version. You cannot undo this change because it is not in conflict with your local version.

Conflicts, identified in the bottom pane by a bounding rectangle, must be resolved manually. To accept the change from one of the two files at the top, do one of the following:

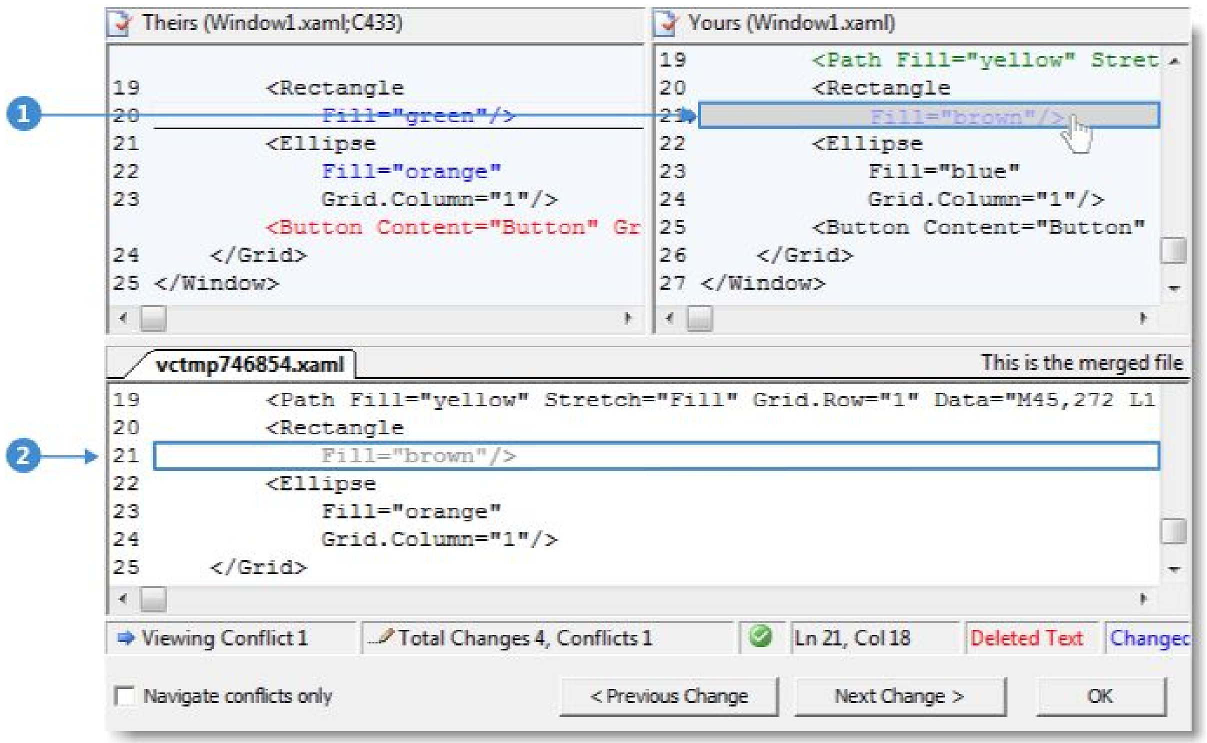
- Click the line whose change you want to apply to the file at the bottom (see the following image).

- Right-click the line whose change you want to apply, and then click **Apply Change**.
- Right-click the original line in the file at the bottom, and then click **Apply Change on the Left** or **Apply Change on the Right**.

To remove a change that you've applied, do one of the following:

- Click a line whose changes have already been applied from one of the files at the top.
- Right-click a line whose changes have already been applied from one of the files at the top, and then click **Remove Change**.
- Right-click the original line in the file at the bottom, and then click **Remove Changes for This Conflict**.

After you apply or remove conflicts from the top two panes, the bottom pane is updated to display the results of the choices that you make.



```
        <Rectangle
            Fill="green"/>
        <Ellipse
            Fill="orange"
            Grid.Column="1"/>
        <Button Content="Button" Gr
    </Grid>
</Window>
```

```
        <Path Fill="yellow" Stret
        <Rectangle
            Fill="brown"/>
        <Ellipse
            Fill="blue"
            Grid.Column="1"/>
        <Button Content="Button"
    </Grid>
</Window>
```

```
        <Path Fill="yellow" Stretch="Fill" Grid.Row="1" Data="M45,272 L1
        <Rectangle
            Fill="brown"/>
        <Ellipse
            Fill="orange"
            Grid.Column="1"/>
    </Grid>
```

1	One of the lines in conflict. Click the line that you want to apply to the merged file.
2	The updated version of the line that is in conflict.

If you **Apply Both Changes**, both lines will appear in the bottom pane, which can cause build errors, but doing so can be useful when you want to compare both lines visually in the bottom pane. You can always remove one of the lines from the bottom pane when you are done comparing.

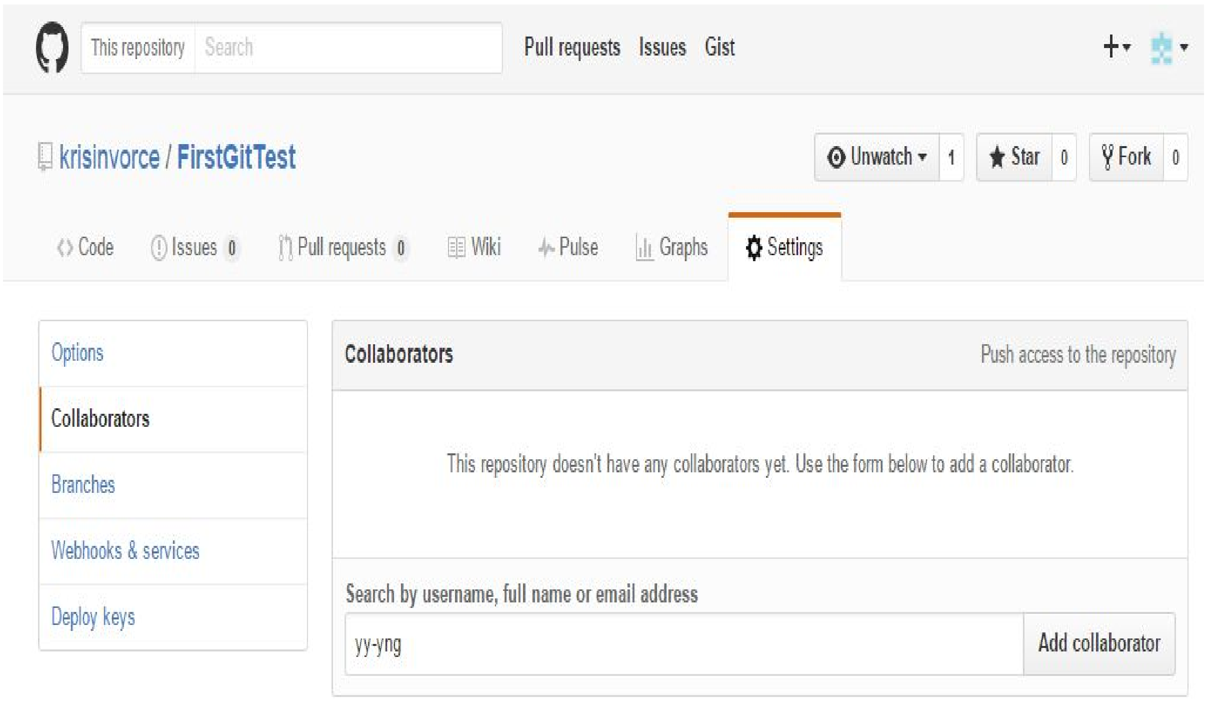
At any time, you can apply all the lines that are in conflict from the server version of the file or from your local file by right-clicking inside the merge tool, pointing to **Resolve All Conflicts**, and then clicking either **With changes on the Left** or **With changes on the Right**. This applies only the lines that are in conflict; it does not remove the changed lines that were not in conflict.
After you click **OK** to exit the merge tool, click **Yes** to save the file, and click **Close** to exit the **Resolve Conflicts** dialog box, remember to test your merged file by building (CTRL+SHIFT+B) or running (F5) your solution before checking the file in.

The importance of merging

Merging is necessary when a file is modified by two people on two different computers at the same time. When two branches are merged, the result is a single collection of files that contains both sets of changes.

In some cases, the merge can be performed automatically, because there is sufficient history information to reconstruct the changes, and the changes do not conflict. In other cases, a person must decide exactly what the resulting files should contain. Many revision control software tools include merge capabilities.

Source Control Management Demonstration(GitHub)


This repository | Search | Pull requests | Issues | Gist

krisinvorce / FirstGitTest

Unwatch 1 | Star 0 | Fork 0

Code | Issues 0 | Pull requests 0 | Wiki | Pulse | Graphs | Settings

Options | Collaborators | Branches | Webhooks & services | Deploy keys

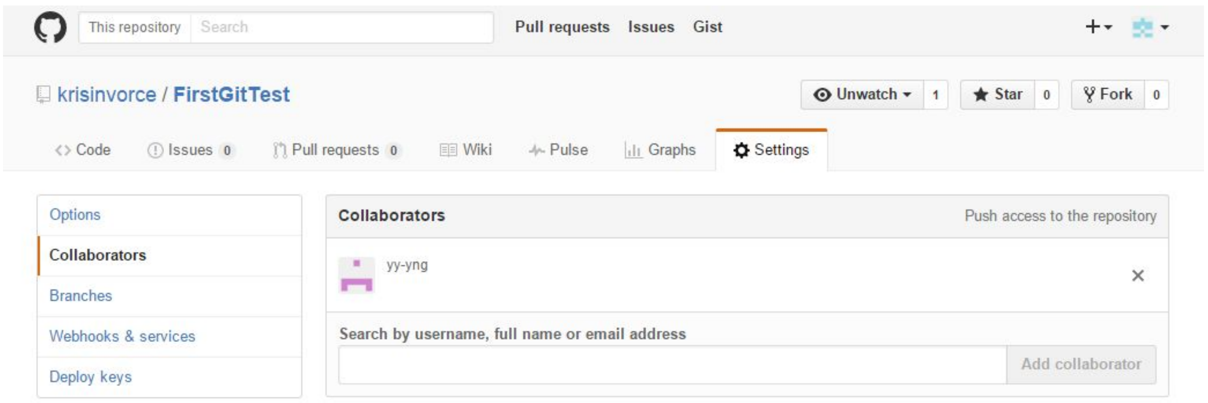
Collaborators — Push access to the repository

This repository doesn't have any collaborators yet. Use the form below to add a collaborator.

Search by username, full name or email address

yy-yng [Add collaborator]

As user krisinvorce, the repository FirstGitTest is accessed. A new collaborator, yy-yng is added.


This repository | Search | Pull requests | Issues | Gist

krisinvorce / FirstGitTest

Unwatch 1 | Star 0 | Fork 0

Code | Issues 0 | Pull requests 0 | Wiki | Pulse | Graphs | Settings

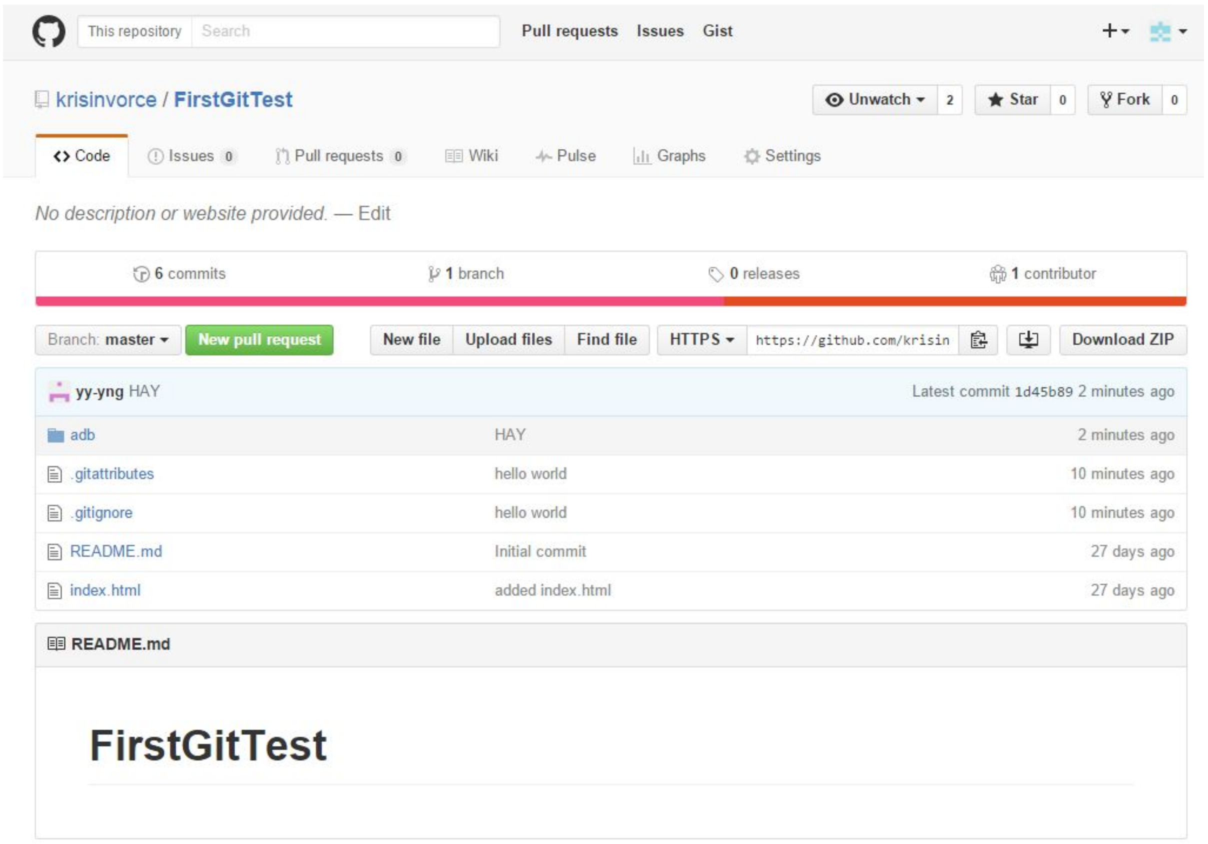
Options | Collaborators | Branches | Webhooks & services | Deploy keys

Collaborators — Push access to the repository

yy-yng ×

Search by username, full name or email address

[Add collaborator]


This repository Search
Pull requests Issues Gist
Unwatch 2 Star 0 Fork 0
Code Issues 0 Pull requests 0 Wiki Pulse Graphs Settings
Branch: master New pull request New file Upload files Find file HTTPS Download ZIP

krisinvorce / FirstGitTest

No description or website provided. — Edit

6 commits · 1 branch · 0 releases · 1 contributor

https://github.com/krisin

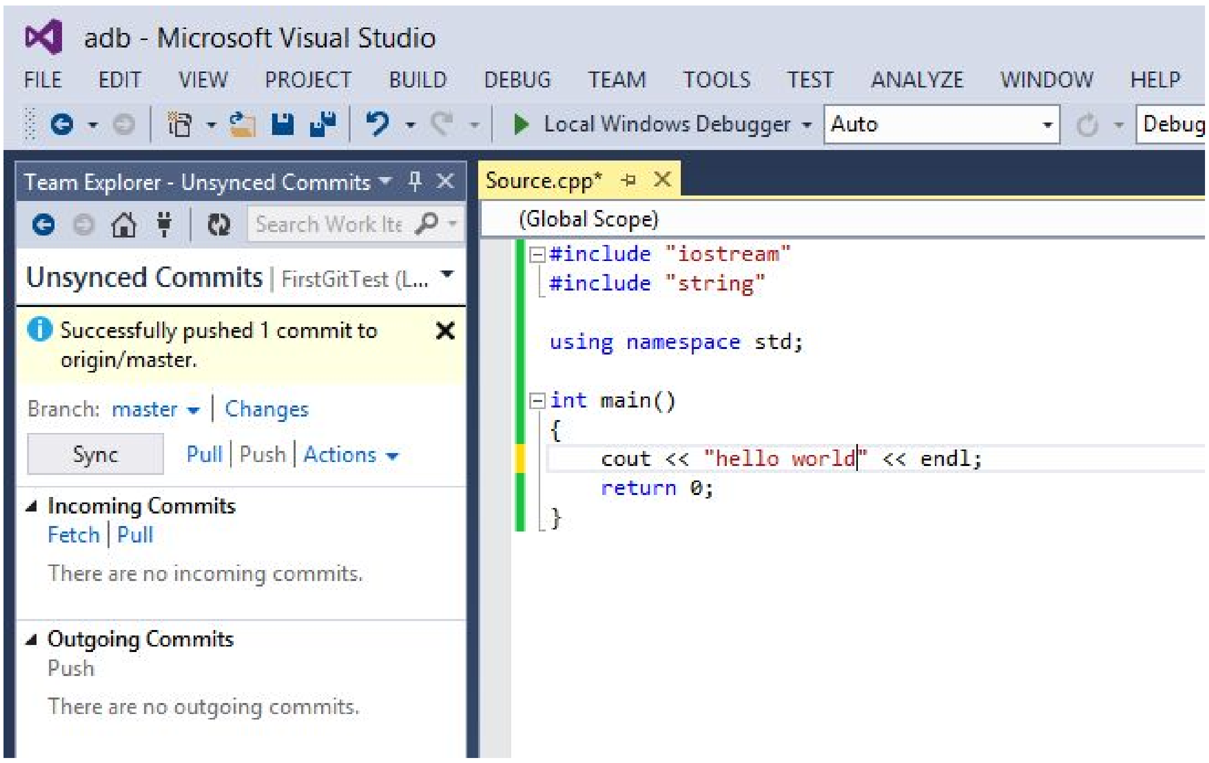
yy-yng HAY — Latest commit 1d45b89 2 minutes ago

adb	HAY	2 minutes ago
.gitattributes	hello world	10 minutes ago
.gitignore	hello world	10 minutes ago
README.md	Initial commit	27 days ago
index.html	added index.html	27 days ago

README.md

FirstGitTest

Collaborator yy-yng commits a new file adb with the message HAY.


adb - Microsoft Visual Studio
FILE EDIT VIEW PROJECT BUILD DEBUG TEAM TOOLS TEST ANALYZE WINDOW HELP
Local Windows Debugger Auto Debug
Team Explorer - Unsynced Commits
Search Work Ite
Source.cpp*
(Global Scope)
Branch: master Changes
Sync Pull Push Actions
Fetch Pull
Push

Unsynced Commits | FirstGitTest (L...

Successfully pushed 1 commit to origin/master.

Incoming Commits

There are no incoming commits.

Outgoing Commits

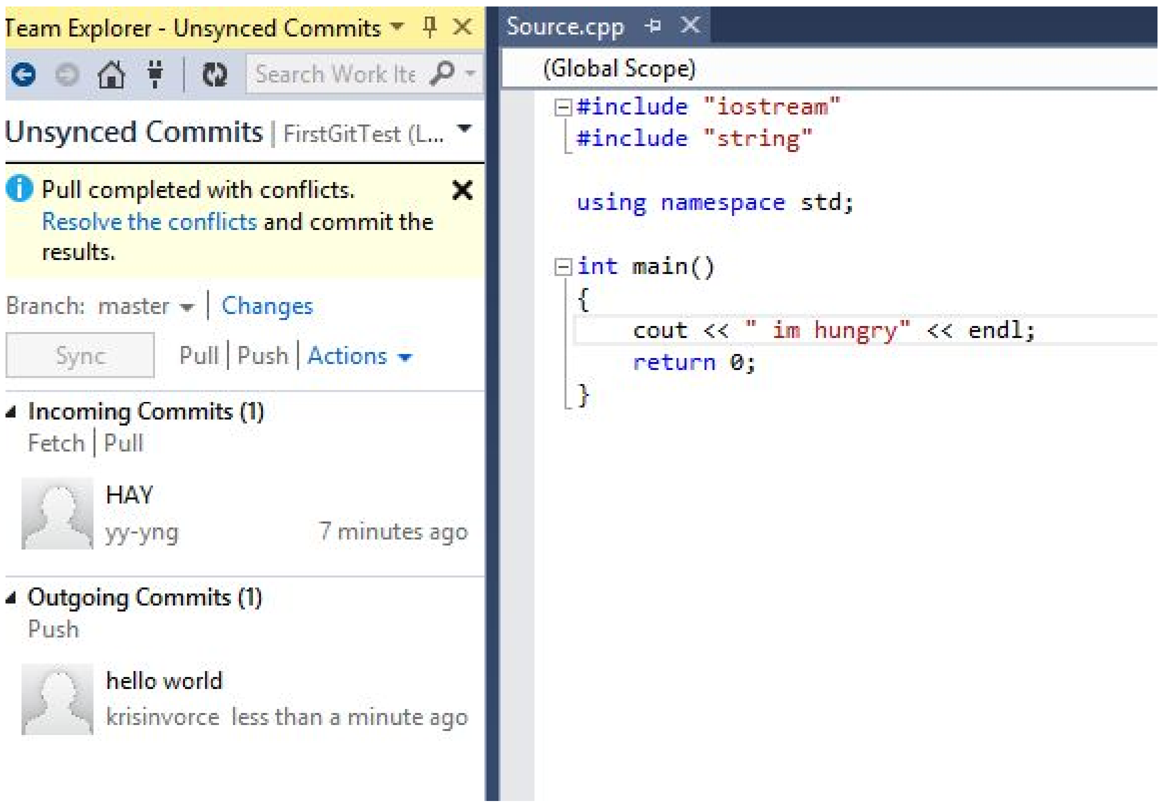
There are no outgoing commits.

```
#include "iostream"
#include "string"

using namespace std;

int main()
{
    cout << "hello world" << endl;
    return 0;
}
```

Attemps are made to alter the content of current file to hello world and how are you doing.



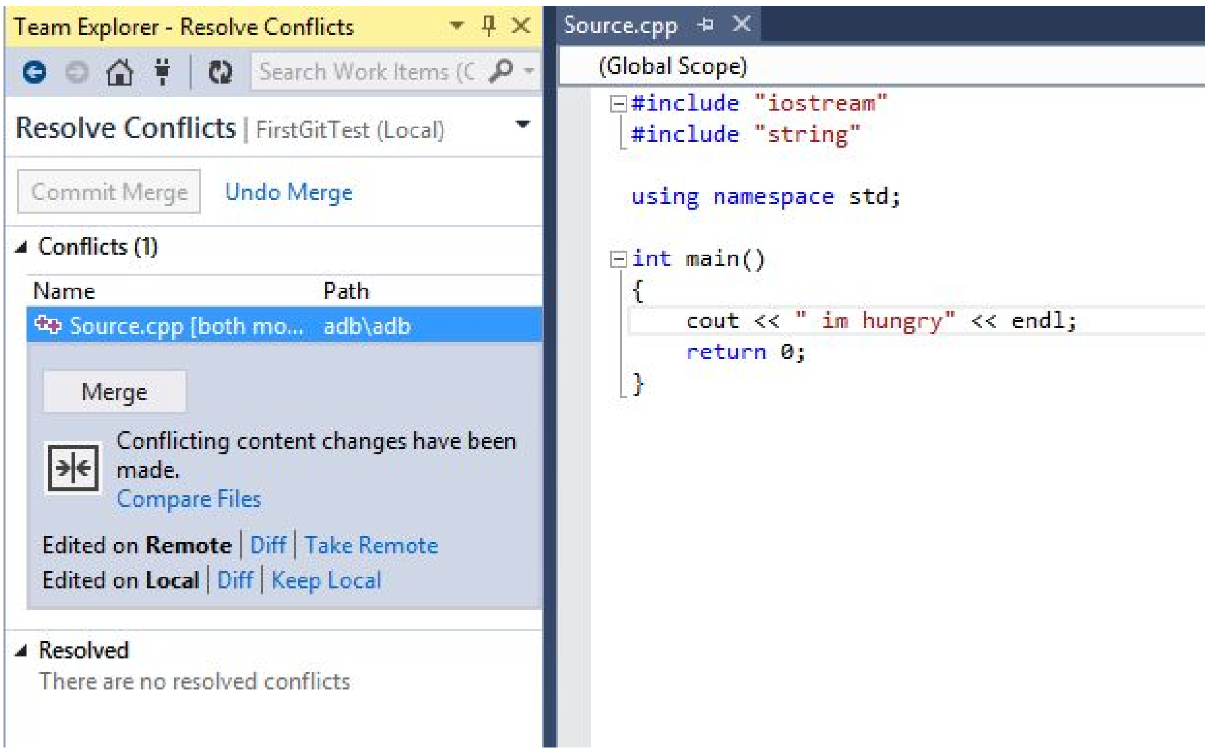
Pull completed with conflicts.
Resolve the conflicts and commit the results.

```
#include "iostream"
#include "string"

using namespace std;

int main()
{
    cout << " im hungry" << endl;
    return 0;
}
```

There is a conflict between the files.


Team Explorer - Resolve Conflicts
Search Work Items (C
Resolve Conflicts | FirstGitTest (Local)
Commit Merge
Undo Merge
Conflicts (1)
Name
Path
Source.cpp [both mo... adb\adb
Merge
Compare Files
Edited on Remote | Diff | Take Remote
Edited on Local | Diff | Keep Local
Resolved
Source.cpp
(Global Scope)

Conflicting content changes have been made.

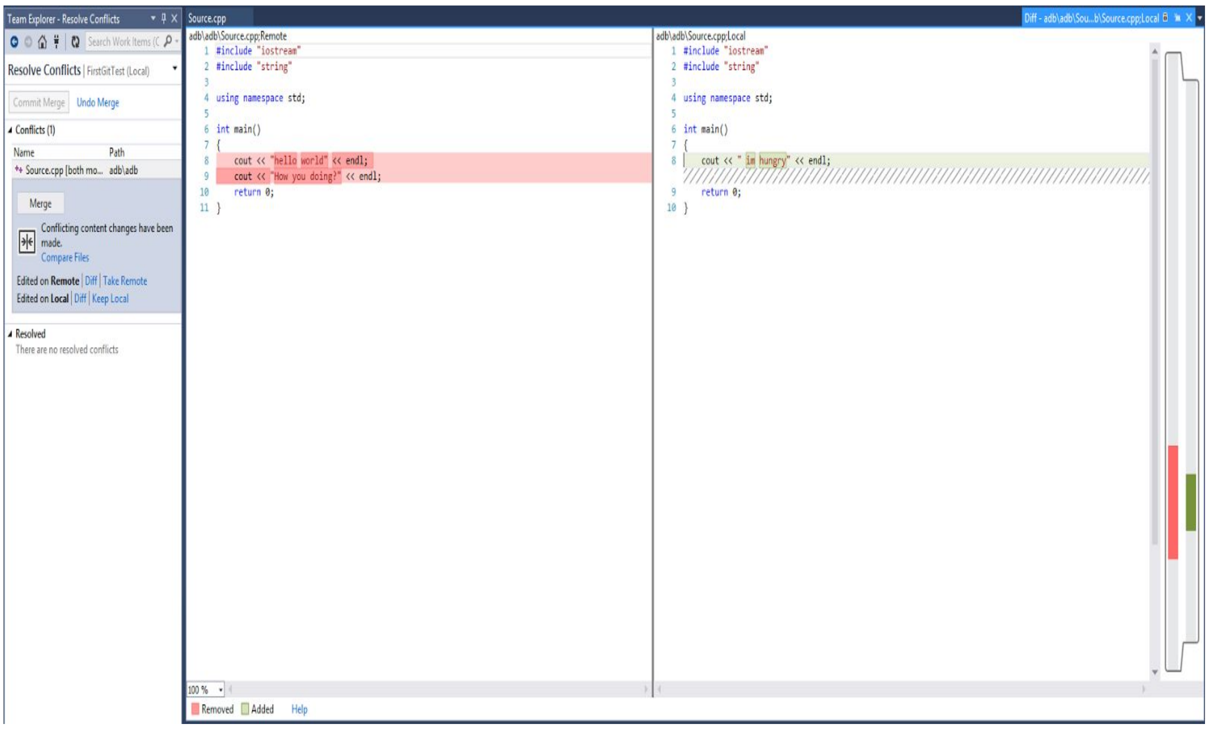
There are no resolved conflicts

```
#include "iostream"
#include "string"

using namespace std;

int main()
{
    cout << " im hungry" << endl;
    return 0;
}
```

Attemps are made to resolve the conflict with both files.



Source.cpp [both mo... adb\adb

Conflicting content changes have been made.

Edited on Remote | Diff | Take Remote

Edited on Local | Diff | Keep Local

There are no resolved conflicts

adb\adb\Source.cpp;Remote

```
#include "iostream"
#include "string"

using namespace std;

int main()
{
    cout << "hello world" << endl;
    cout << "How you doing?" << endl;
    return 0;
}
```

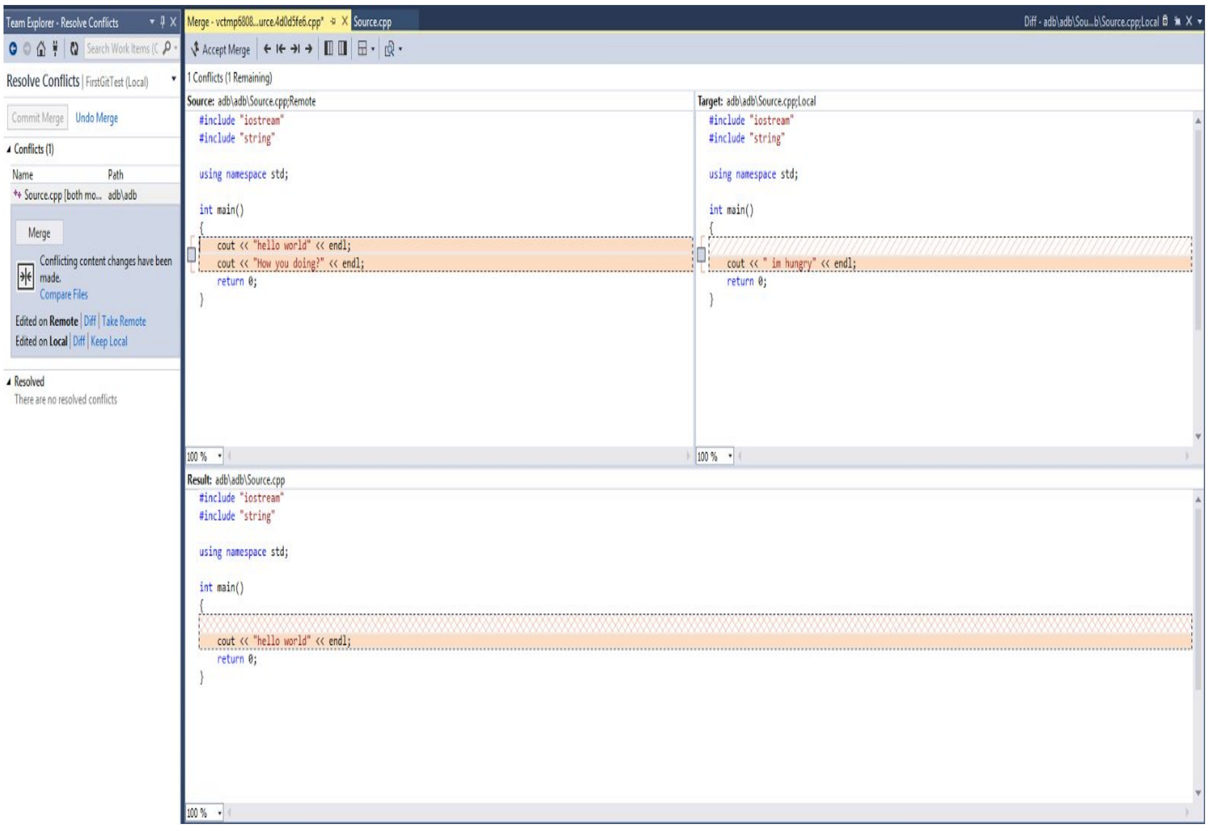
adb\adb\Source.cpp;Local

```
#include "iostream"
#include "string"

using namespace std;

int main()
{
    cout << " im hungry" << endl;
    return 0;
}
```

Comparison between both files are made and attempts are made to resolve conflicts.


Team Explorer - Resolve Conflicts
Search Work Items
Resolve Conflicts | FirstGitTest (Local)
Commit Merge
Undo Merge
Conflicts (1)
Name
Path
Source.cpp
Merge
Compare Files
Resolved
Merge - vctmp6808...urce.4d0d5fe6.cpp*
Source.cpp
Diff - adb\adb\Sou...b\Source.cpp;Local
Accept Merge
100 %

Source.cpp [both mo... adb\adb

Conflicting content changes have been made.

Edited on Remote | Diff | Take Remote

Edited on Local | Diff | Keep Local

There are no resolved conflicts

1 Conflicts (1 Remaining)

Source: adb\adb\Source.cpp;Remote

```
#include "iostream"
#include "string"

using namespace std;

int main()
{
    cout << "hello world" << endl;
    cout << "How you doing?" << endl;
    return 0;
}
```

Target: adb\adb\Source.cpp;Local

```
#include "iostream"
#include "string"

using namespace std;

int main()
{
    cout << " im hungry" << endl;
    return 0;
}
```

Result: adb\adb\Source.cpp

```
#include "iostream"
#include "string"

using namespace std;

int main()
{
    cout << "hello world" << endl;
    return 0;
}
```

A choice is made between the two files with different contents. The final version of the file is displayed at the bottom window panel. Once we are satisfied with the choice, we choose "Attempt merge". Then, the file is pushed to the repository.