Advanced Software Development: Git Workflows

CPSC 233: Introduction to Computer Science for Computer Science Majors II
Winter 2025

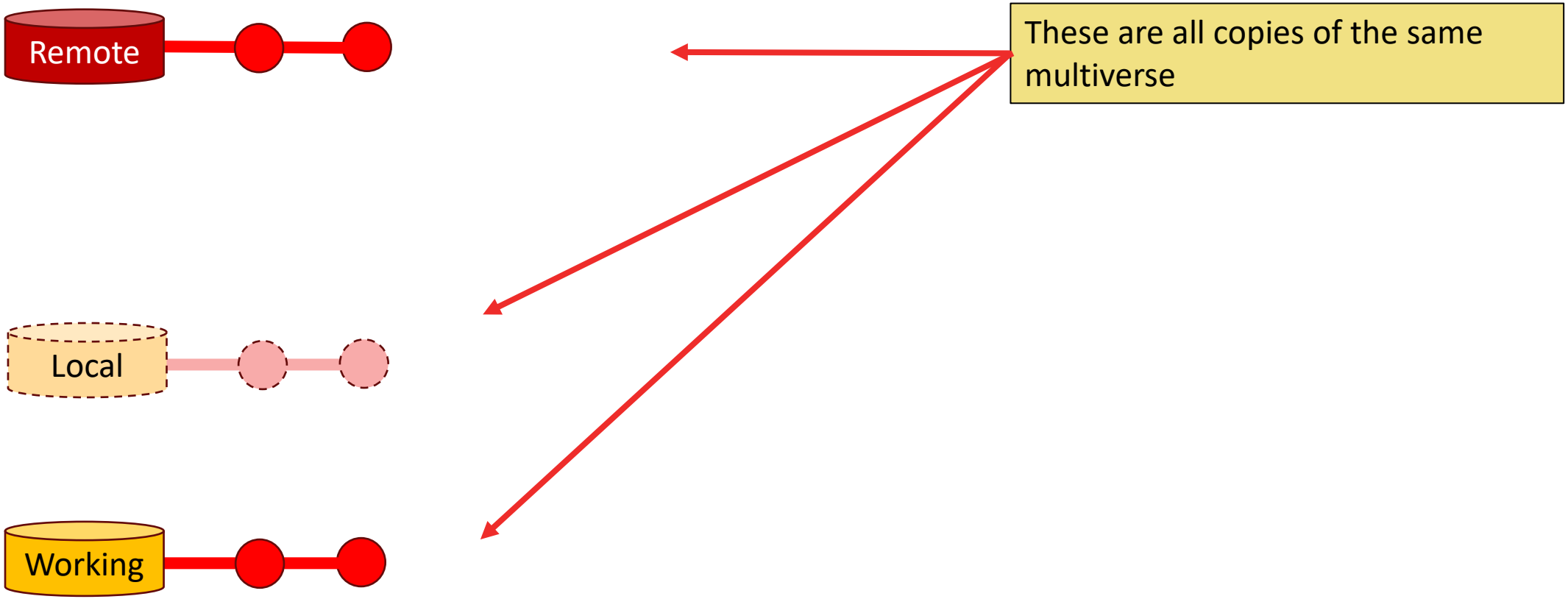
Slides via Steve Sutcliffe

Jonathan Hudson, Ph.D.
Assistant Professor (Teaching)
Department of Computer Science
University of Calgary

Wednesday, January 8, 2025

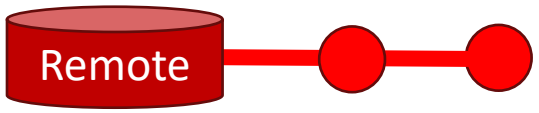
UNIVERSITY OF CALGARY

Workflow: the three realms

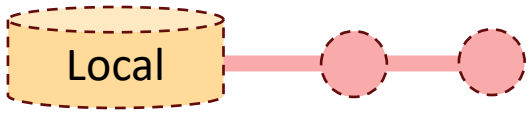


UNIVERSITY OF CALGARY

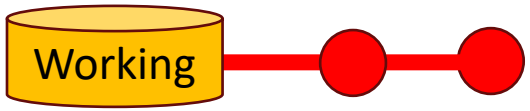
Workflow: the three realms



This is the version the world sees; all magicians have access to this one

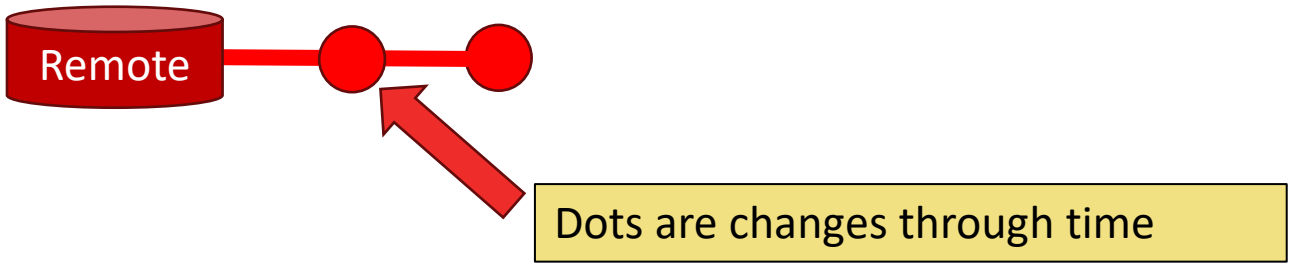


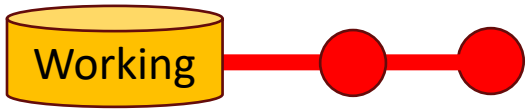
This is the version only Barry has access to, it is **git's local copy** of the "remote", it is invisible to Barry.



This is the version Barry works on

Workflow: the three realms

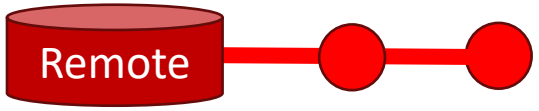


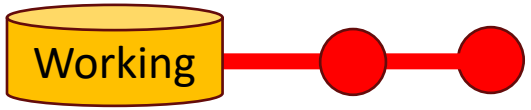


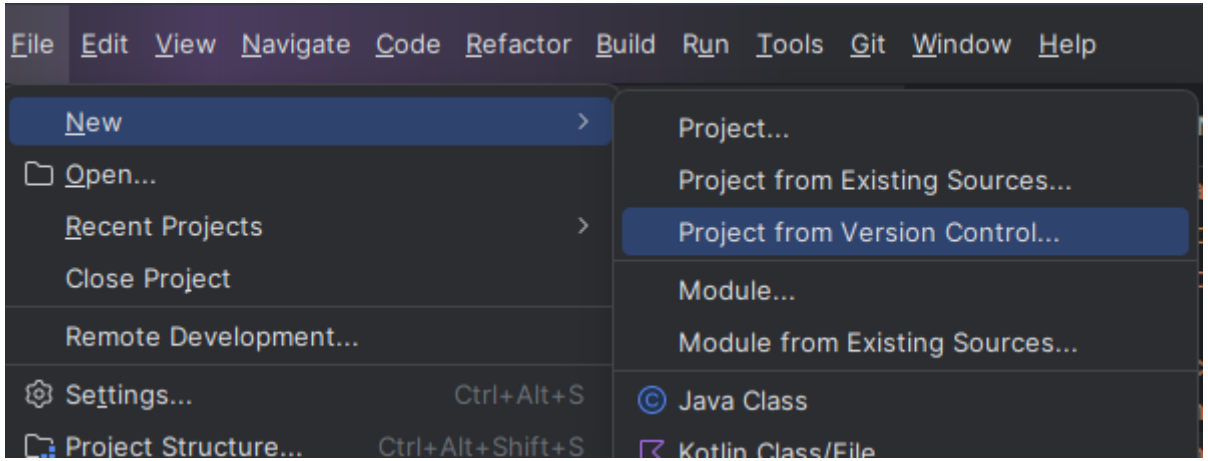
Solo Work

Clone

Workflow: Clone the Repo



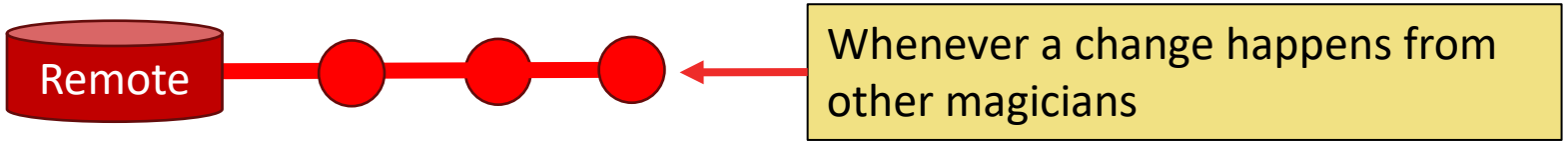


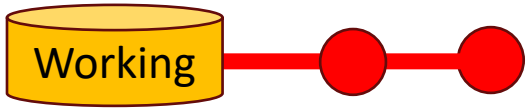


Barry creates this when he clones the repo down to his local machine

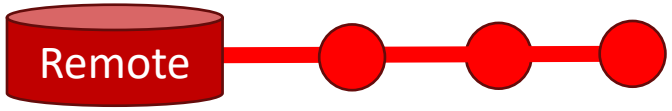
Pull

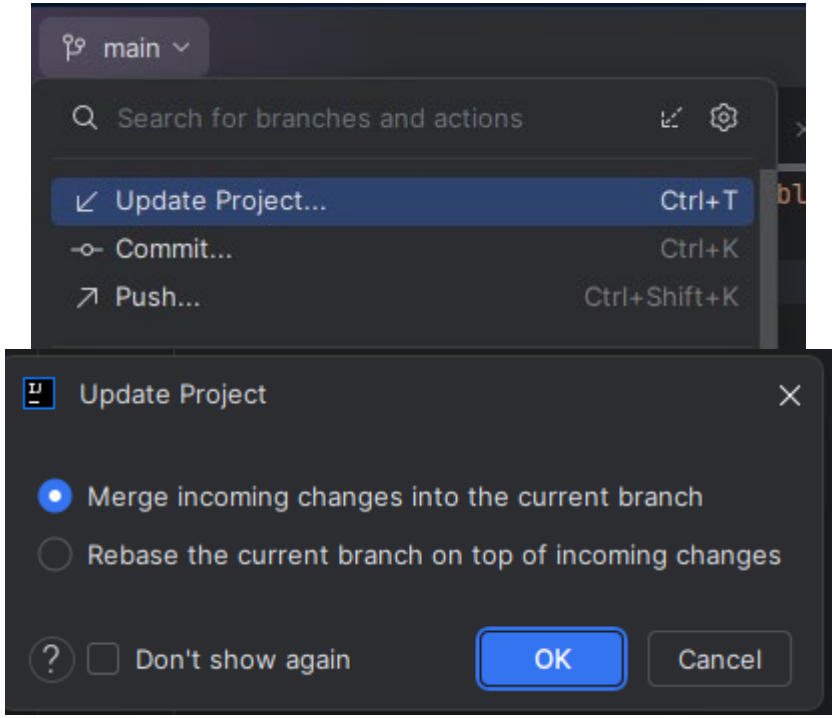
Workflow: pull (update in intellij)

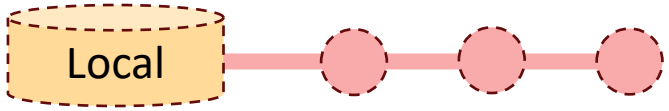




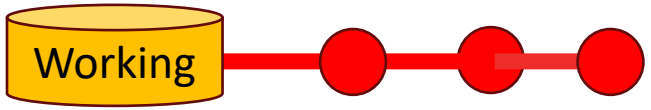
Workflow: pull (update in intellij)







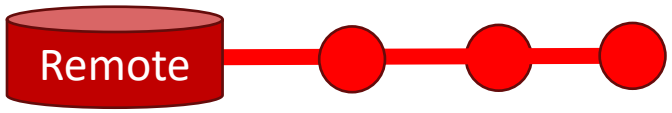
Barry must update his version by pulling those changes from the remote

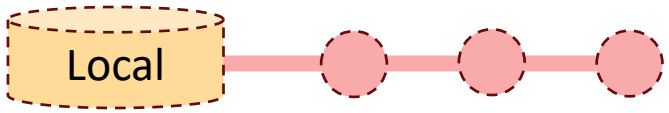


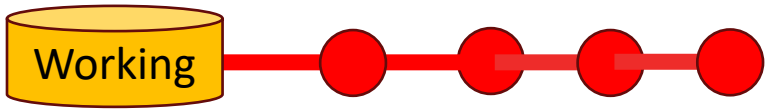
That ‘pull’ updates the area where Barry works.

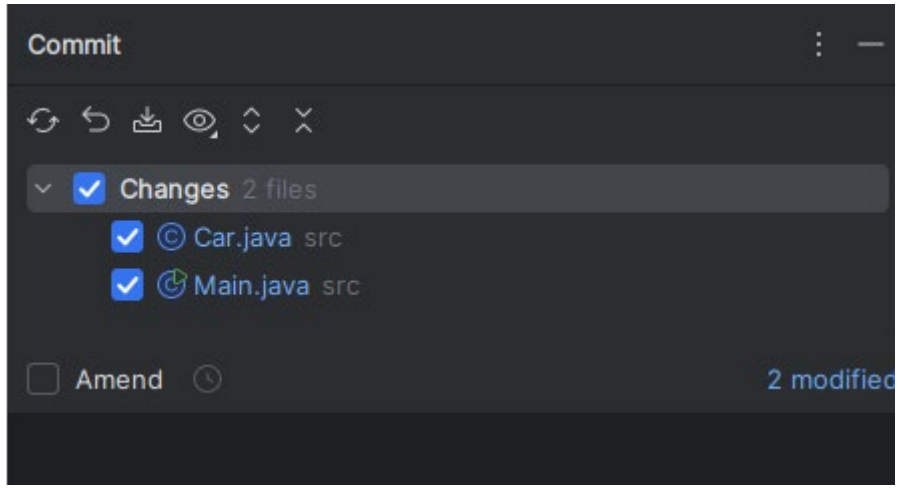
Work

Workflow: add (checkboxes in intellij)







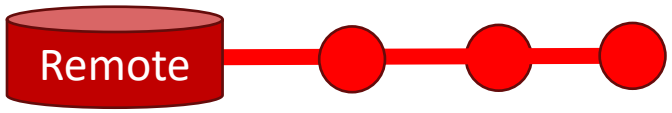


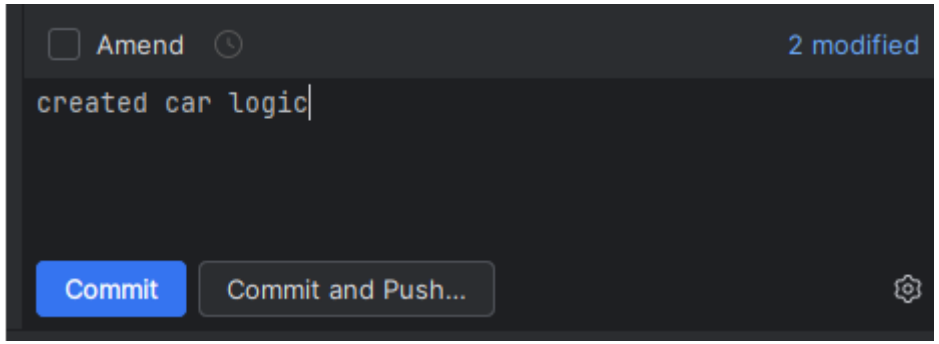
When Barry does something, a change in time happens only for his working

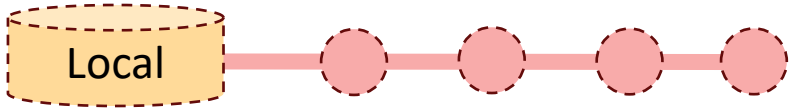
UNIVERSITY OF CALGARY

Save

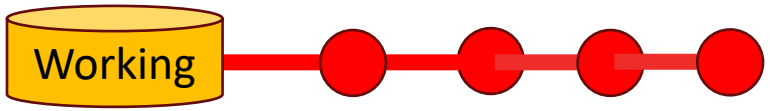
Workflow: Commit





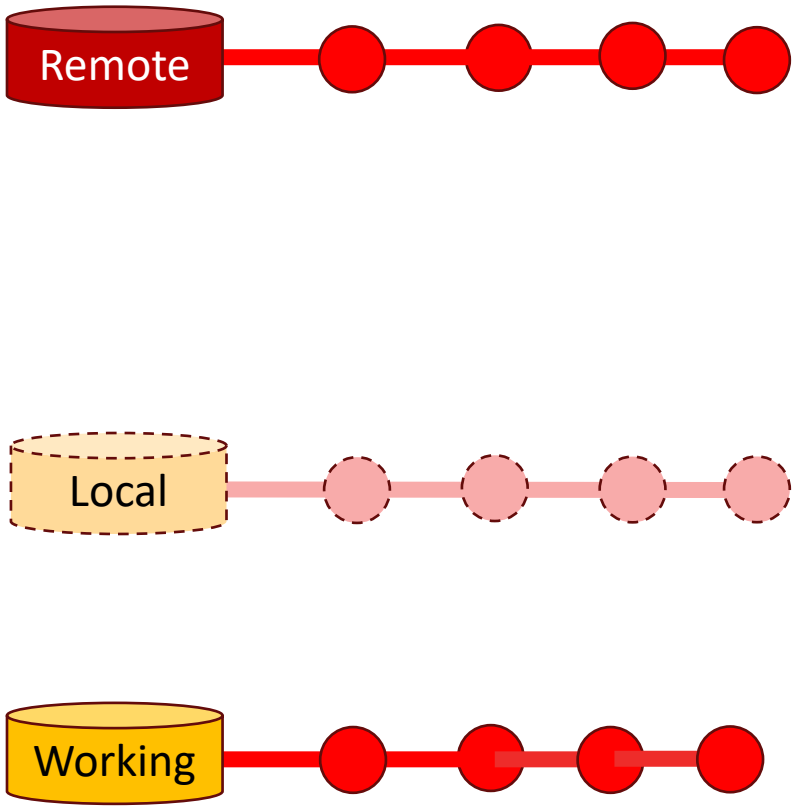


If Barry is happy with that change, he needs to commit to it

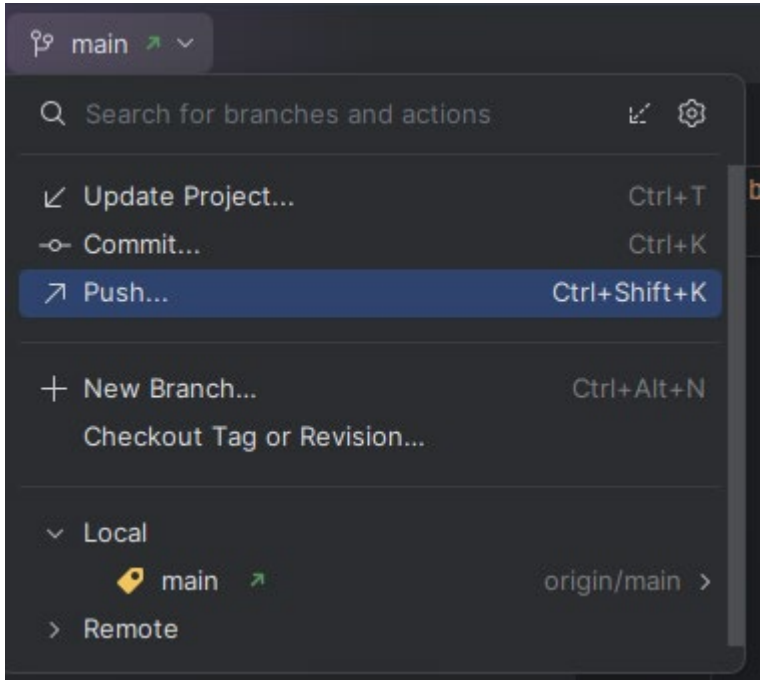


Push

Workflow: Push



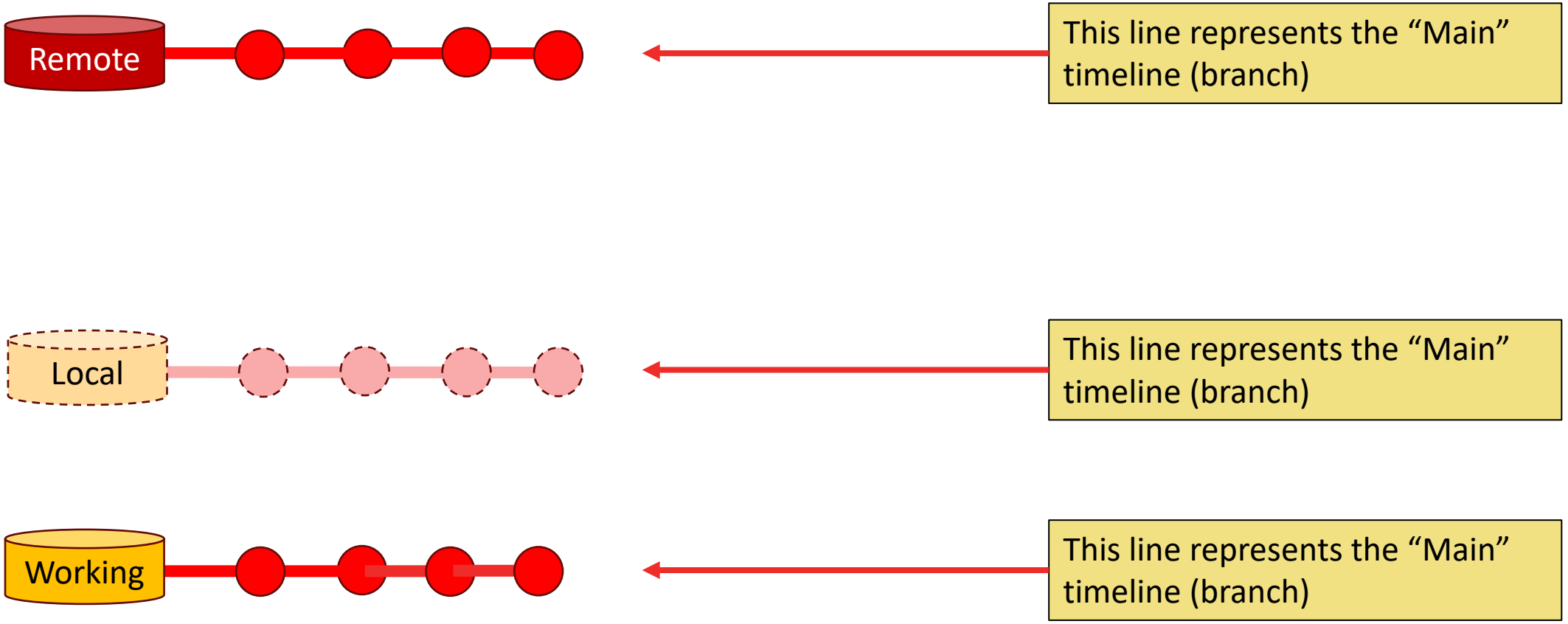
If Barry wants to share that change with other magicians, he must push those changes to the world



Branches

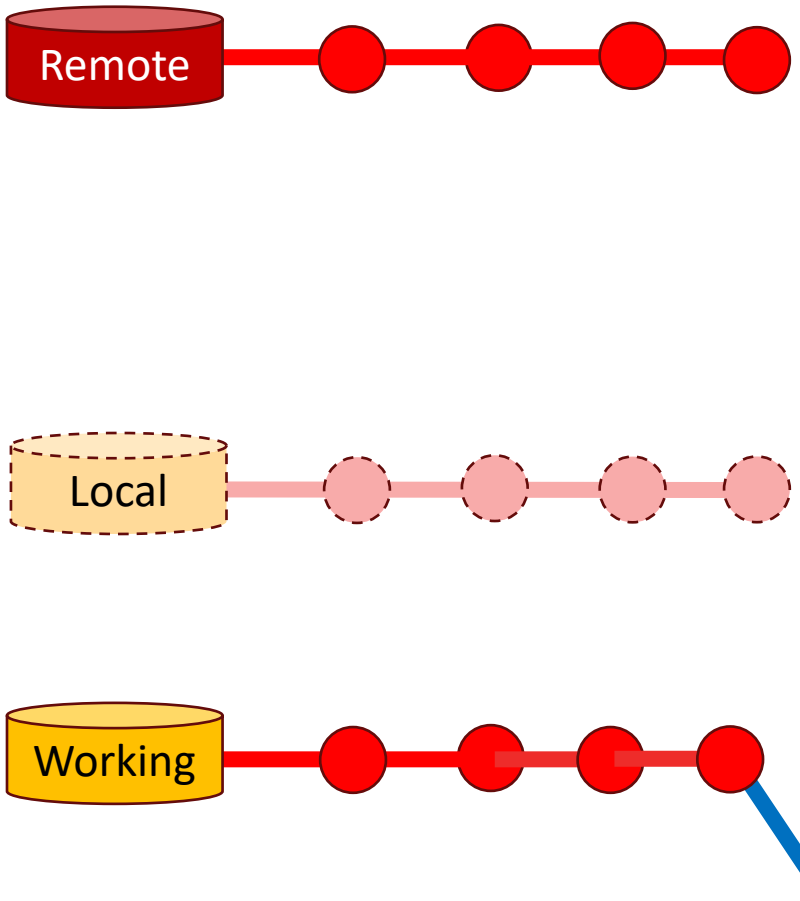
Main

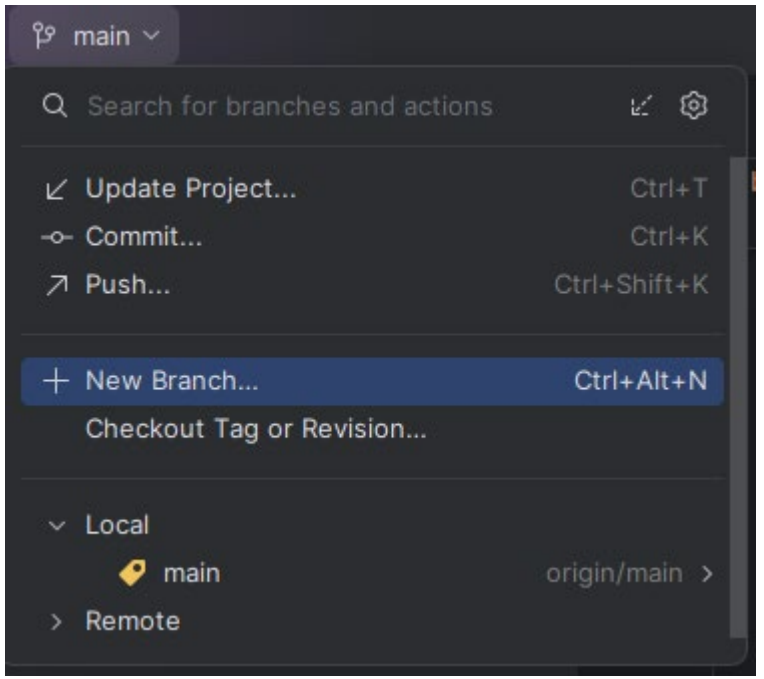
Workflow: Branches -1-



They are just different areas for working on the main timeline

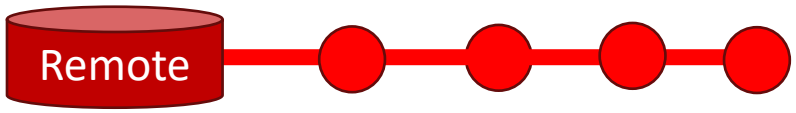
Workflow: Branches -2-

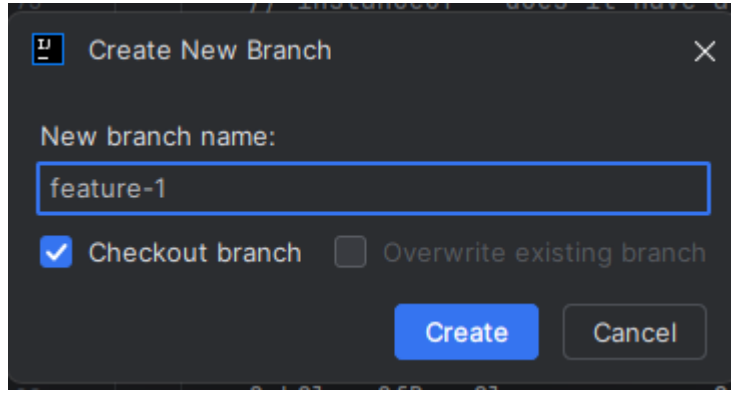


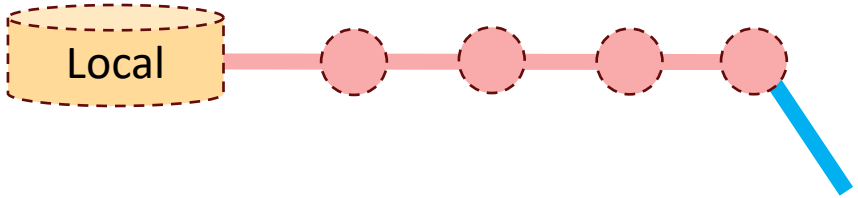


If Barry wants to experiment, he creates a new timeline (branch)

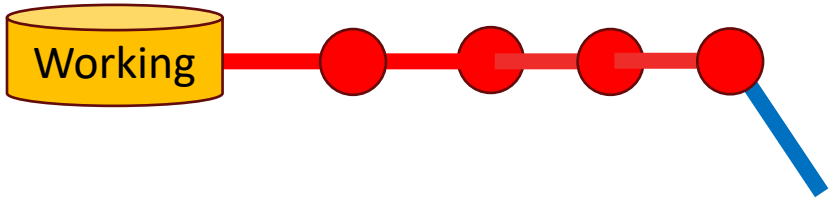
Workflow: Branches -2-



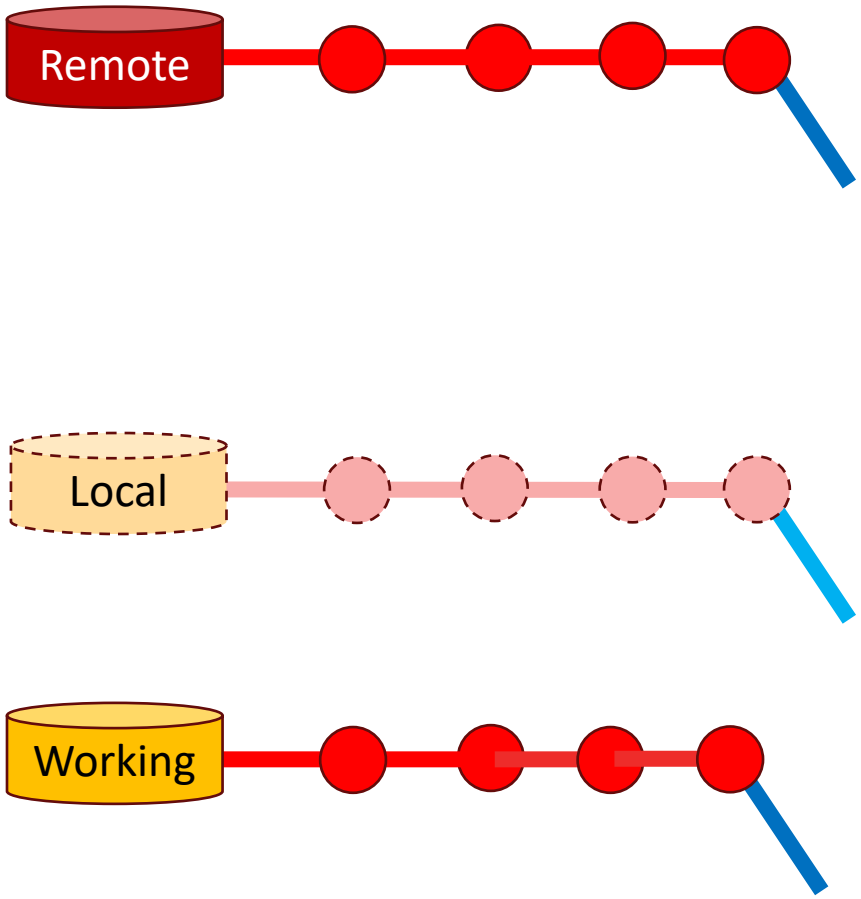




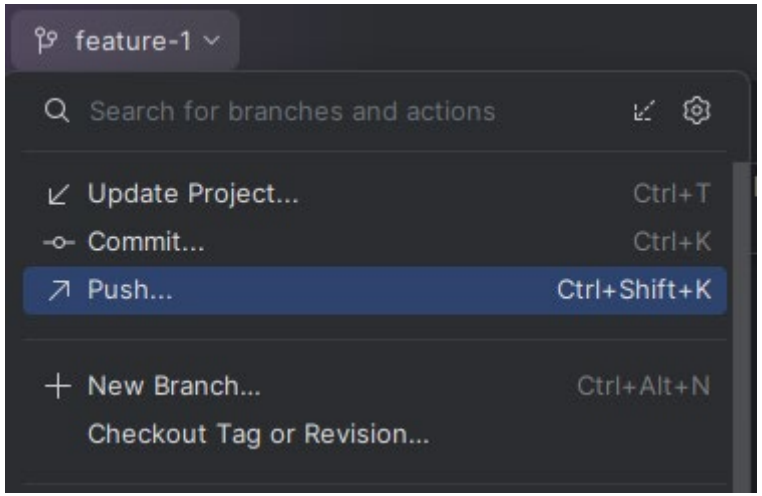
Barry's copy of the multiverse is automatically updated



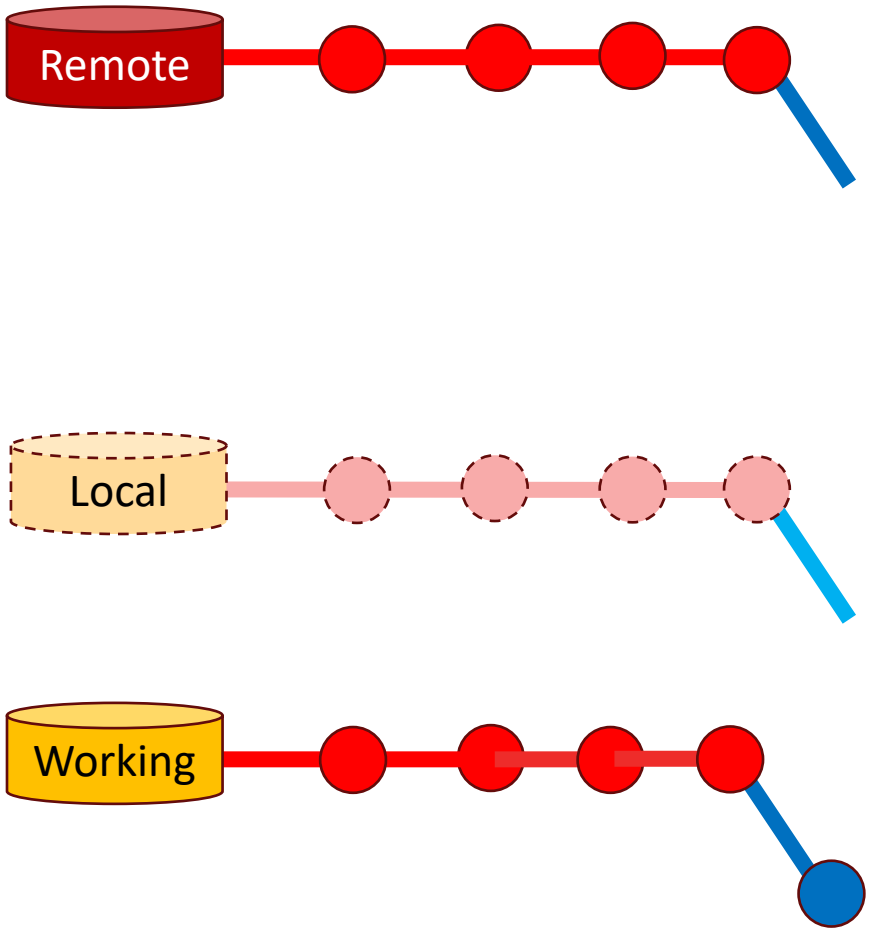
Workflow: Branches -2- (push in intellij)

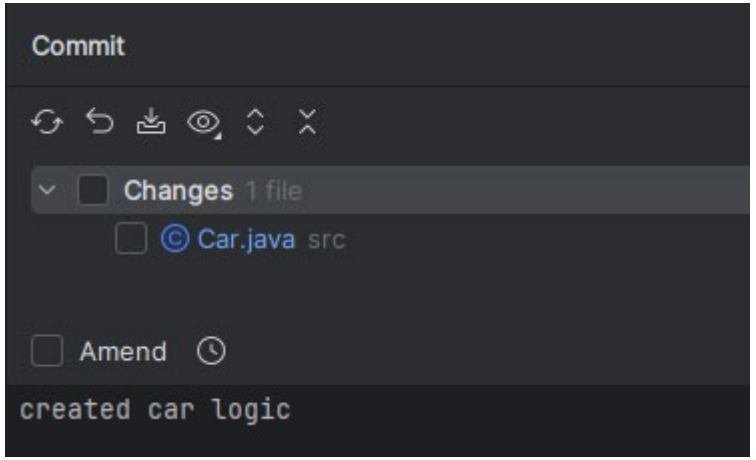


But if Barry wants other magicians to see this branching timeline, he must set the upstream



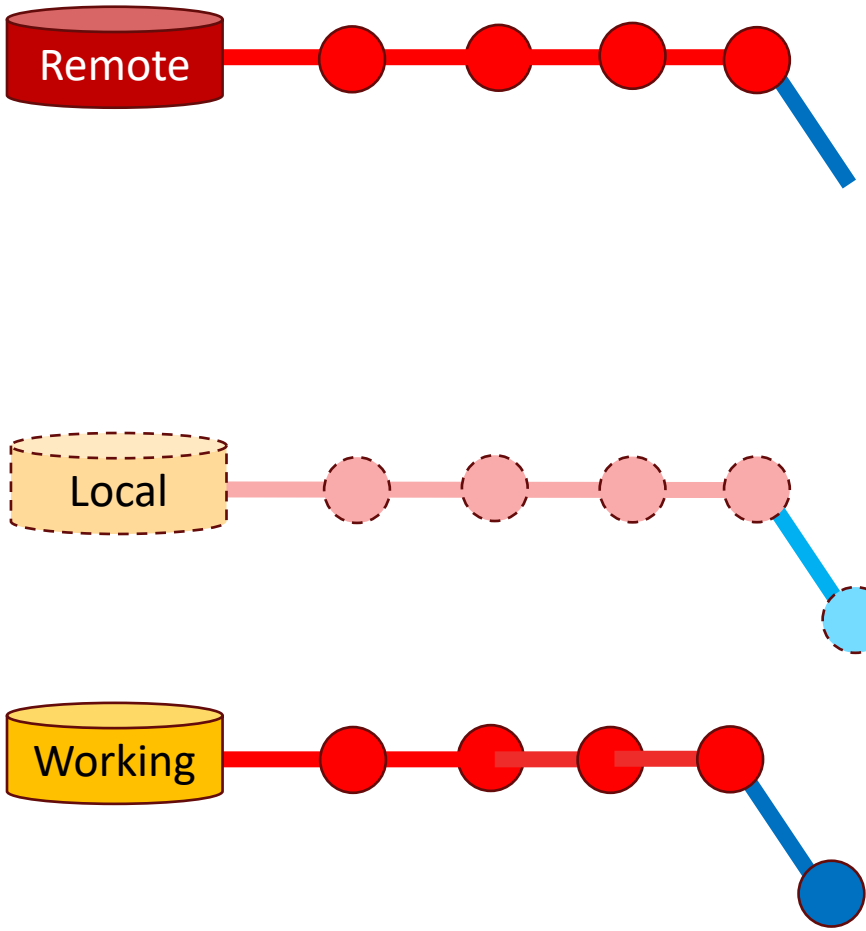
Workflow: Branches -3-

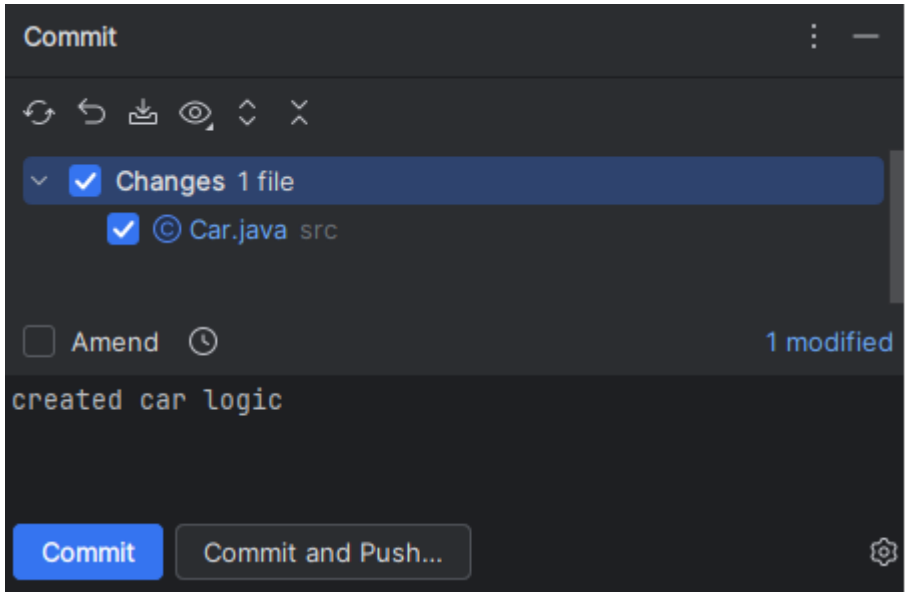




Now when Barry does something, it only effects this new branch, in the working area

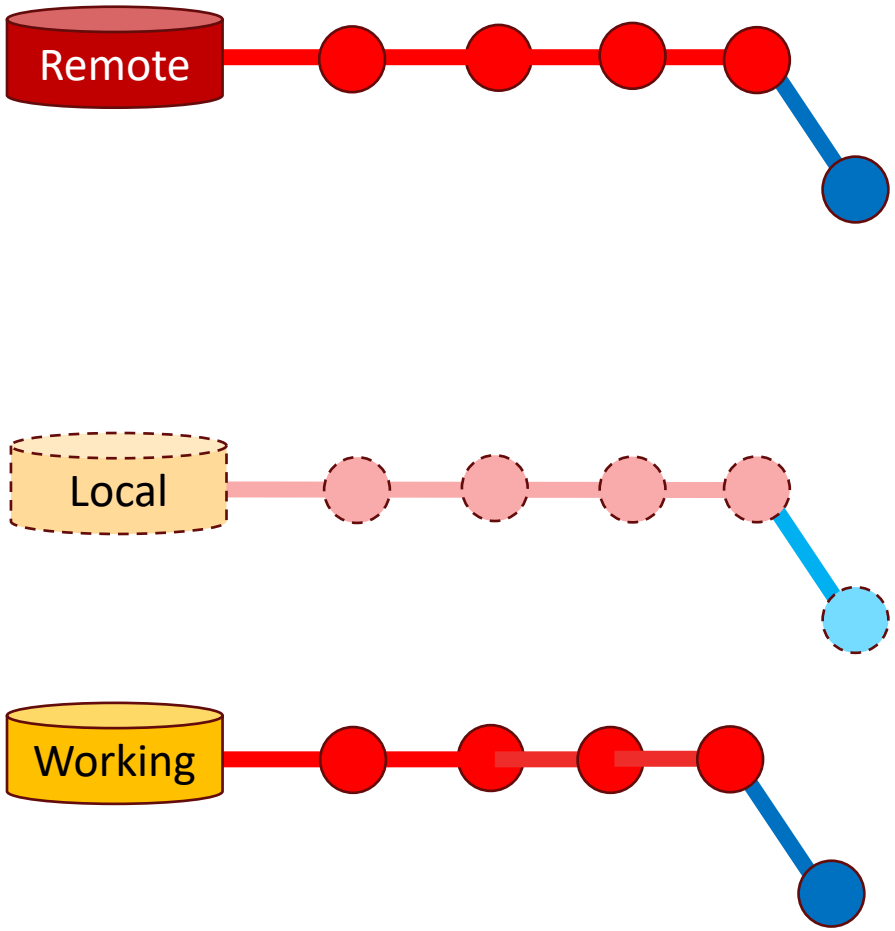
Workflow: Branches -3-



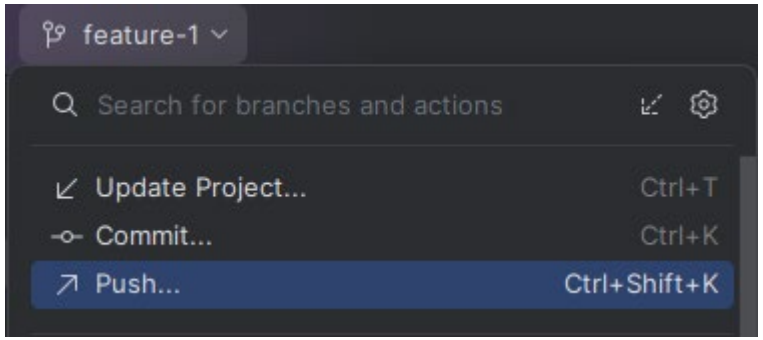


If Barry is happy with that change, he needs to “add” the changes and “commit” it.

Workflow: Branches -3-

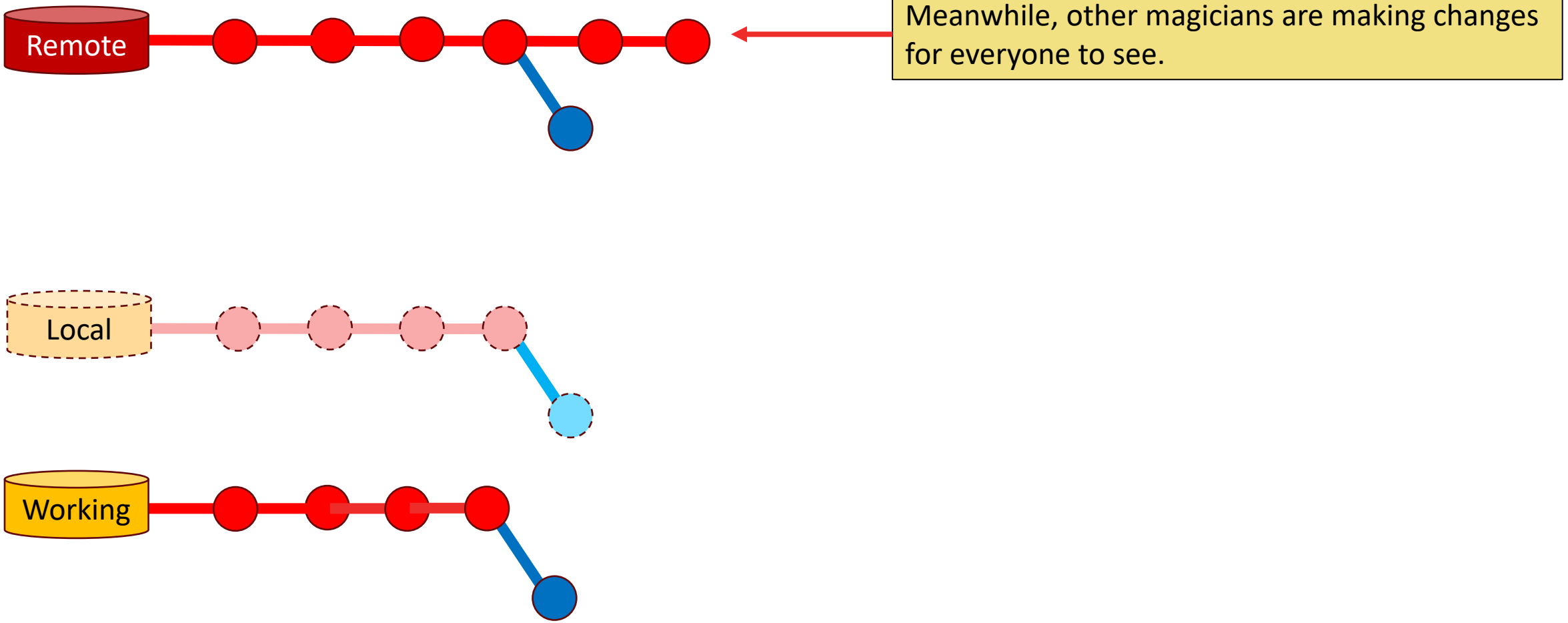


If he wants other magicians to access it, he must “push” those changes to the remote.

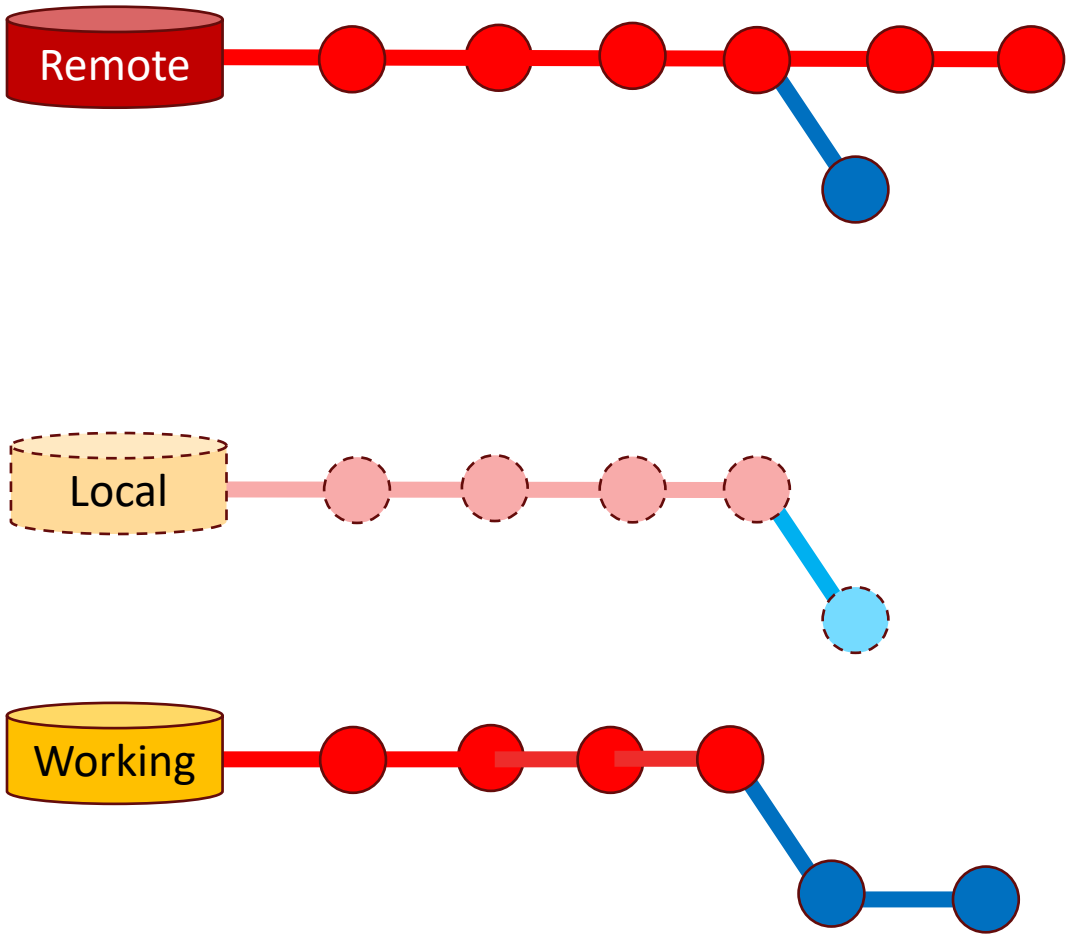


Experiment/Feature/Fix

Workflow: Merge -1-

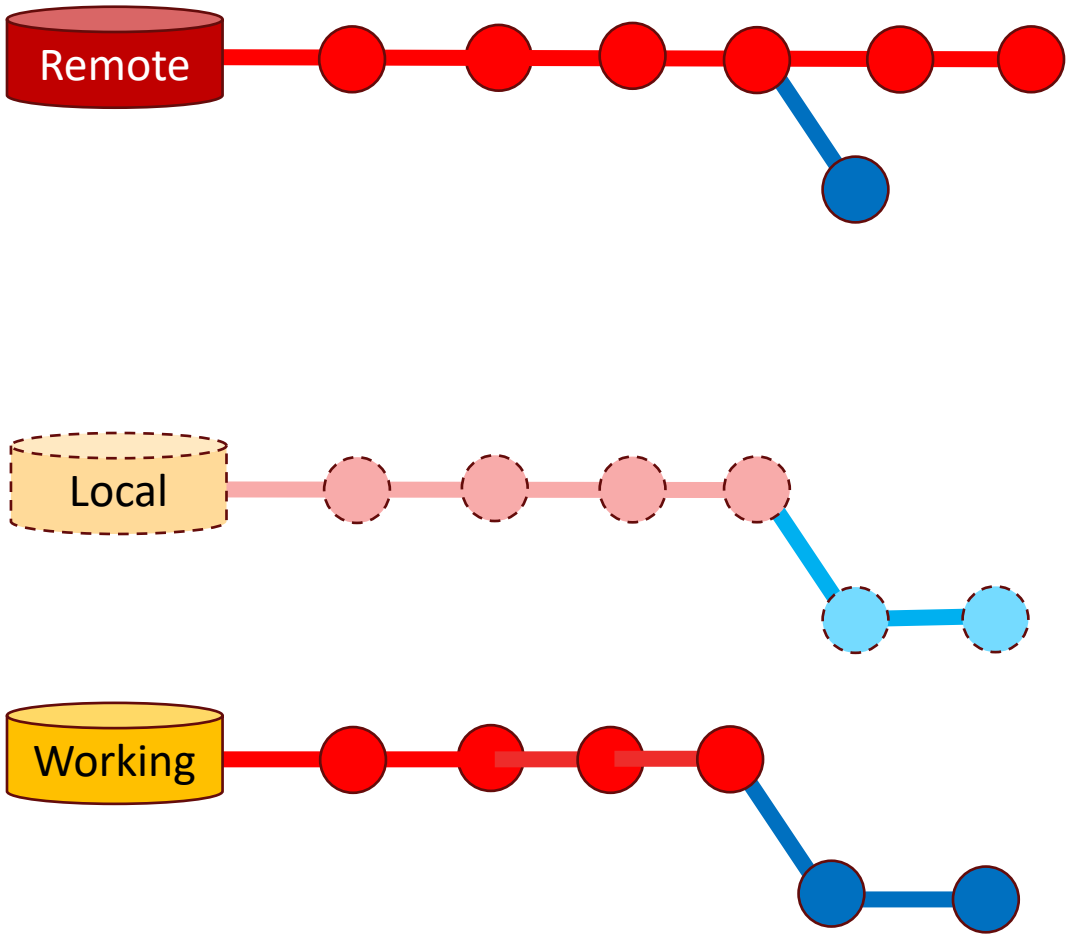


Workflow: Merge -1-



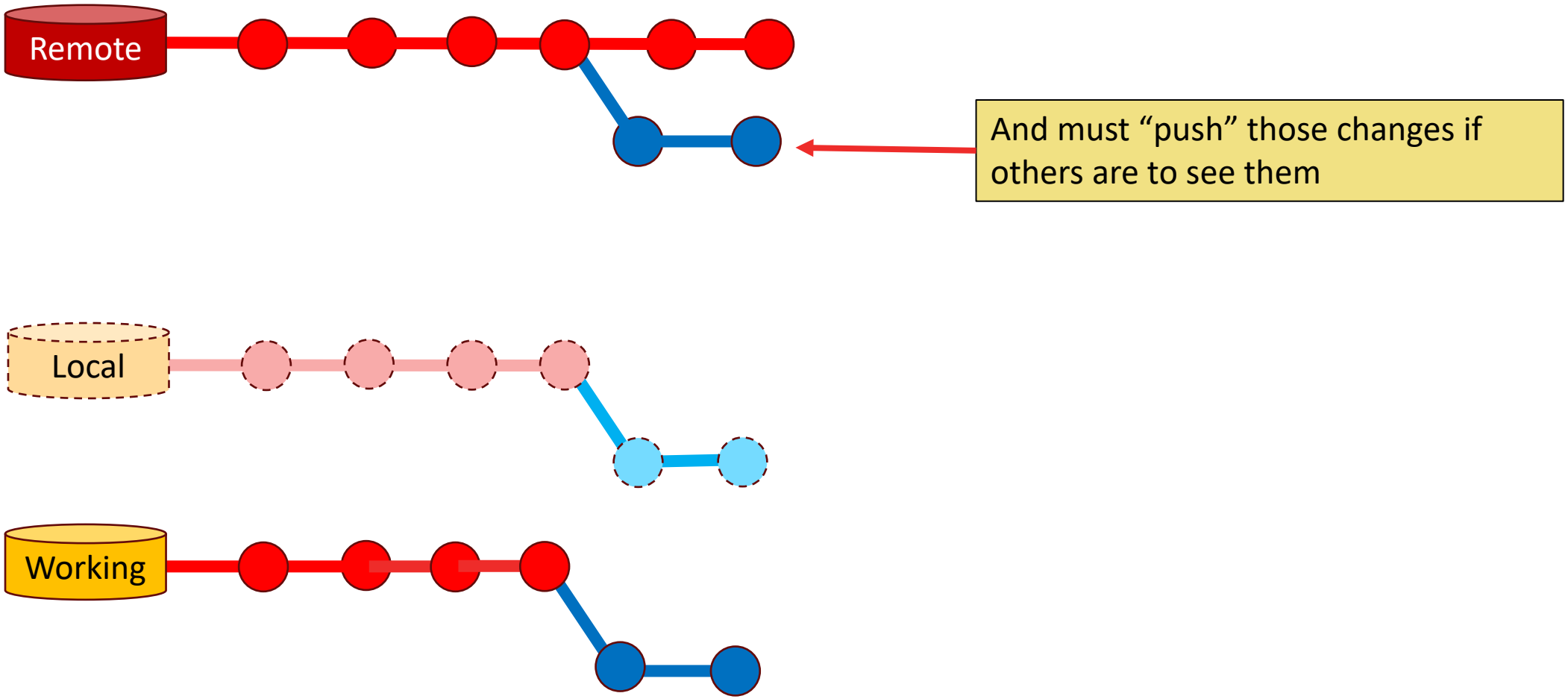
Barry can keep making his own changes.

Workflow: Merge -1-



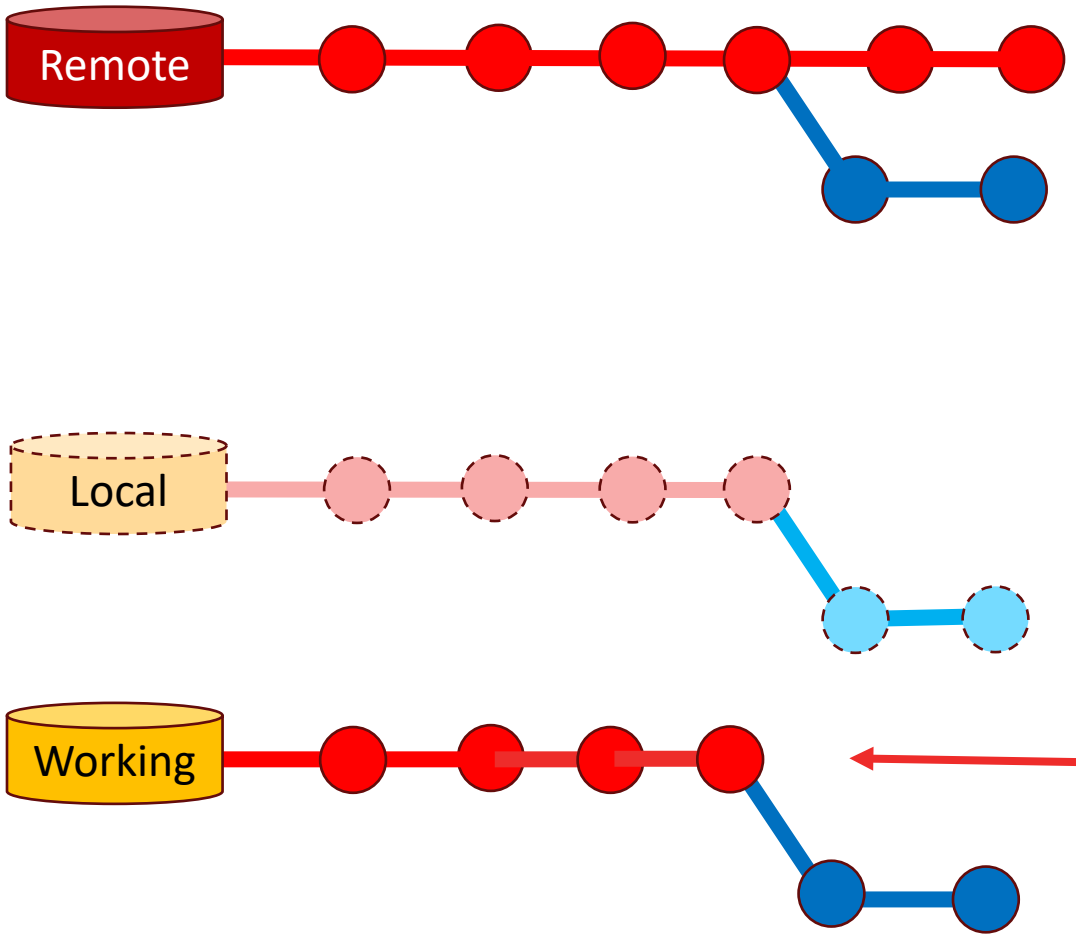
And committing those changes

Workflow: Merge -1-



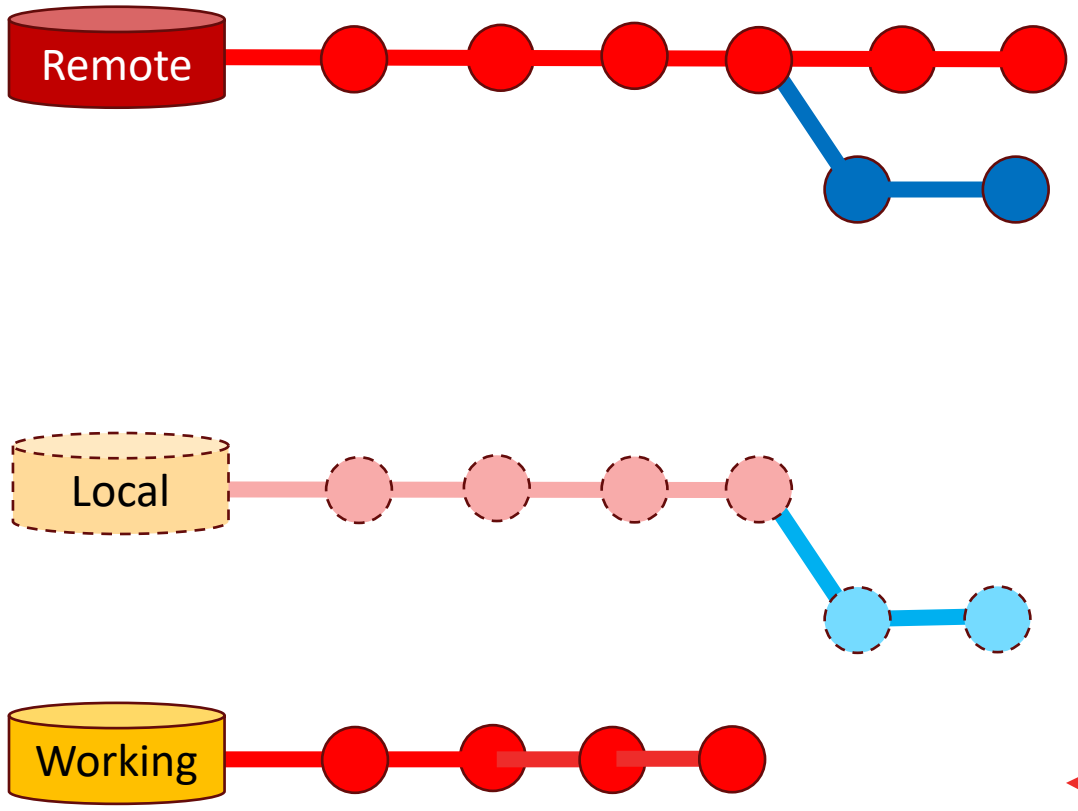
Merge

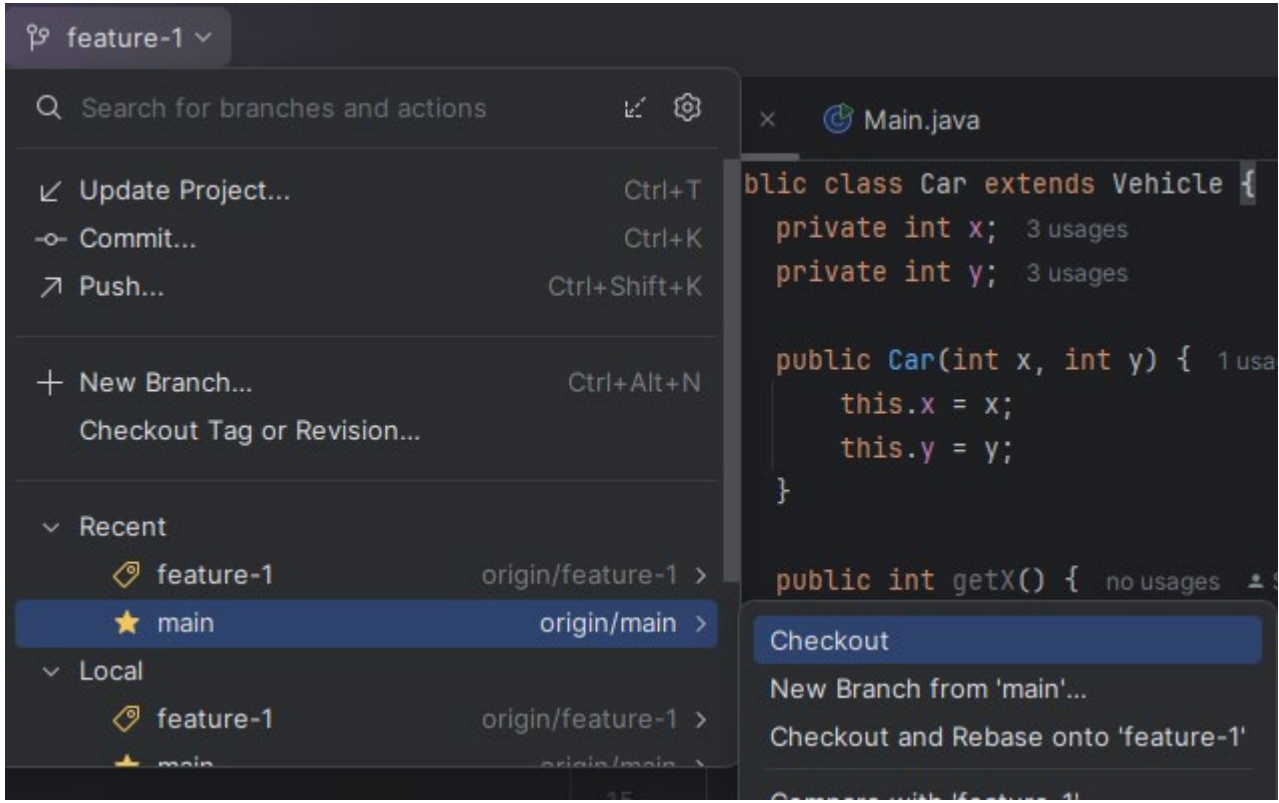
Workflow: Merge -1-



But notice, Barry's version of the main timeline is not up to date!

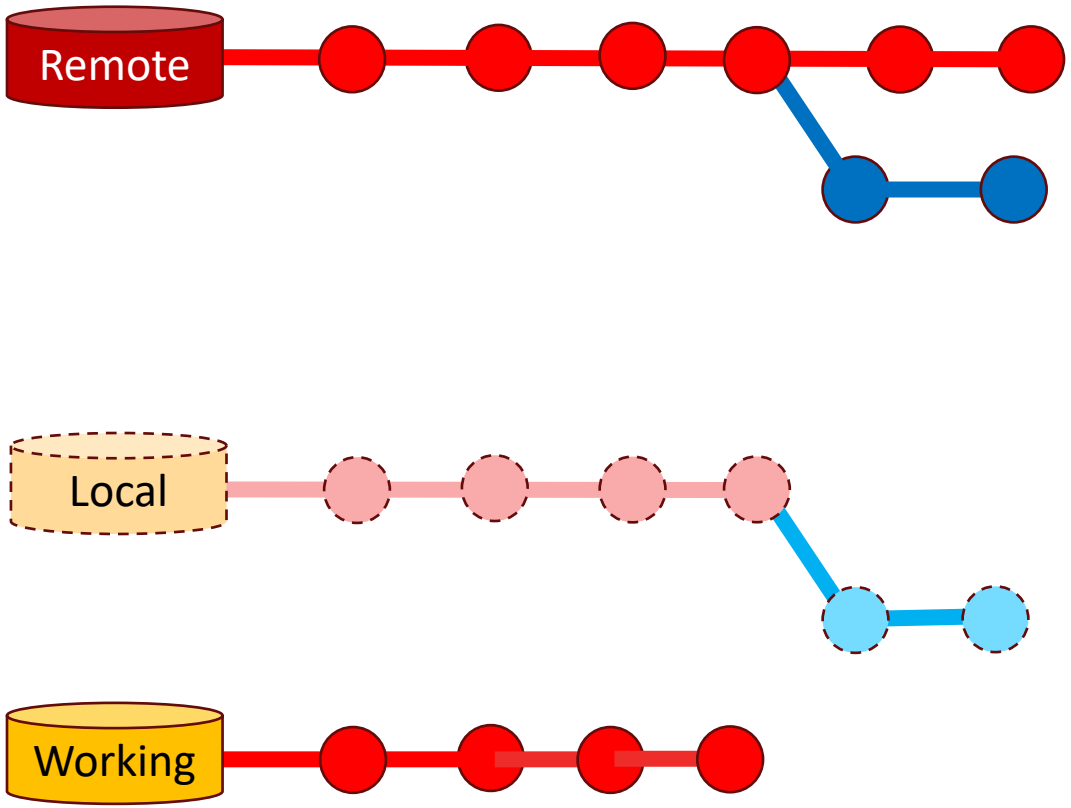
Workflow: Merge -1-





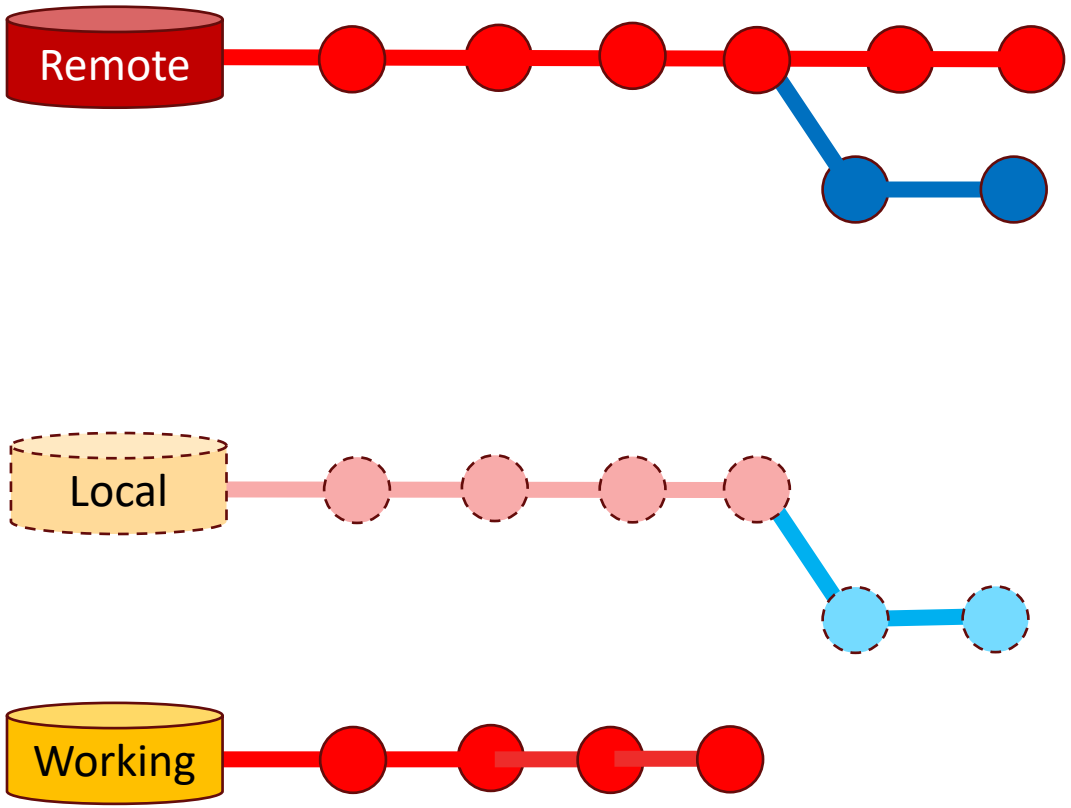
To update his version, he must "checkout" the main branch

Workflow: Merge -1-



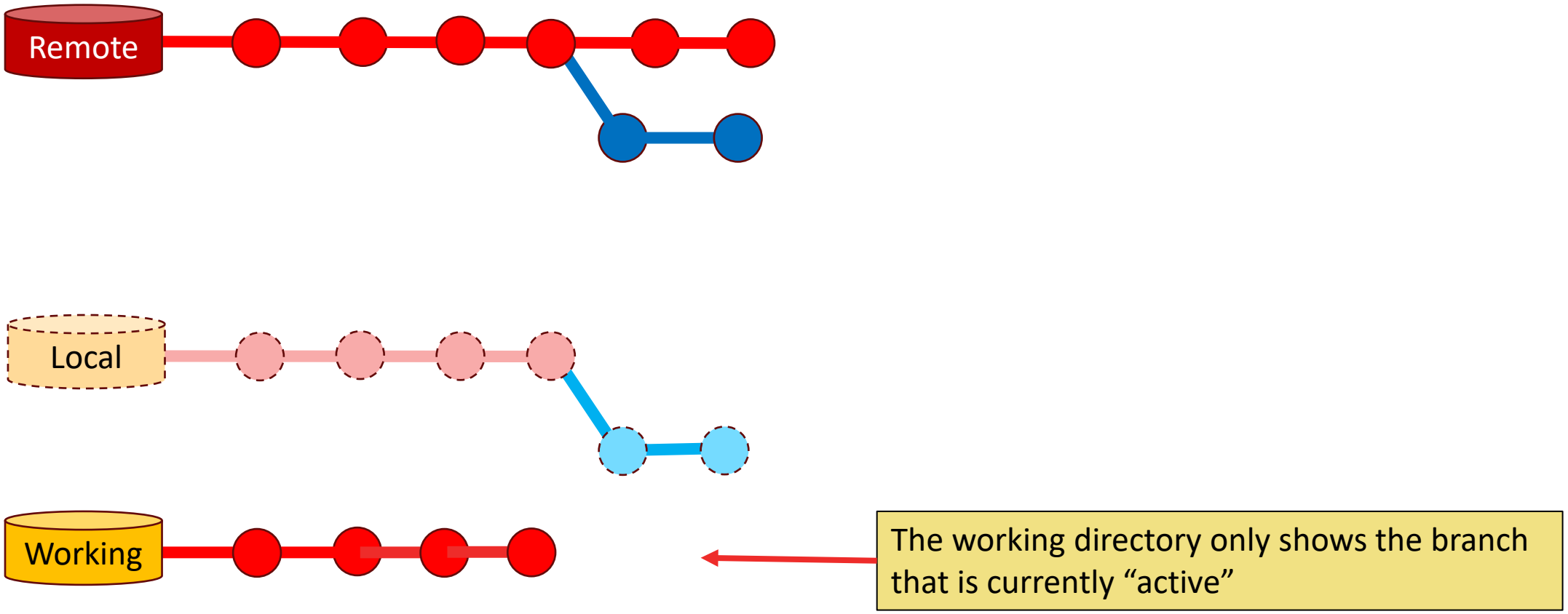
But where did all of his branch changes go?

Workflow: Merge -1-

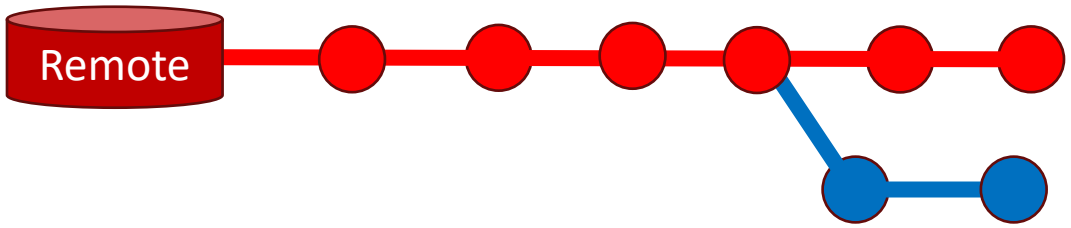


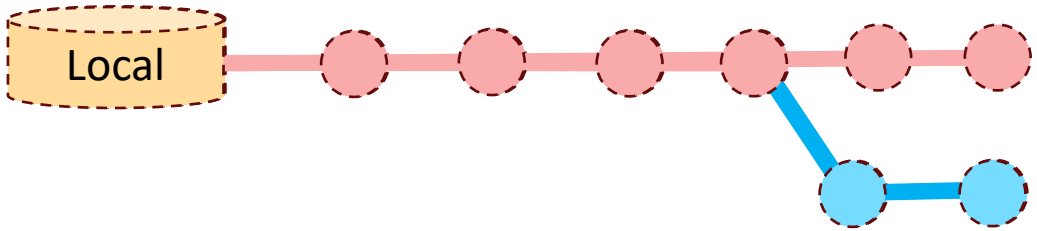
They are still preserved here, as long as he has been “committing” the changes

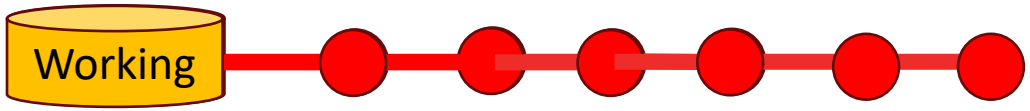
Workflow: Merge -1-

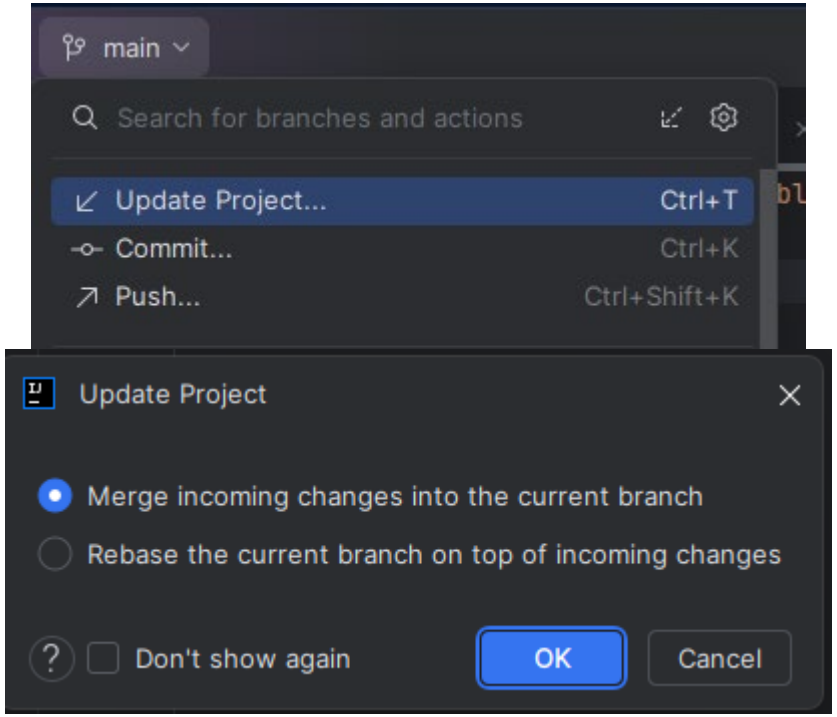


Workflow: Merge -2-





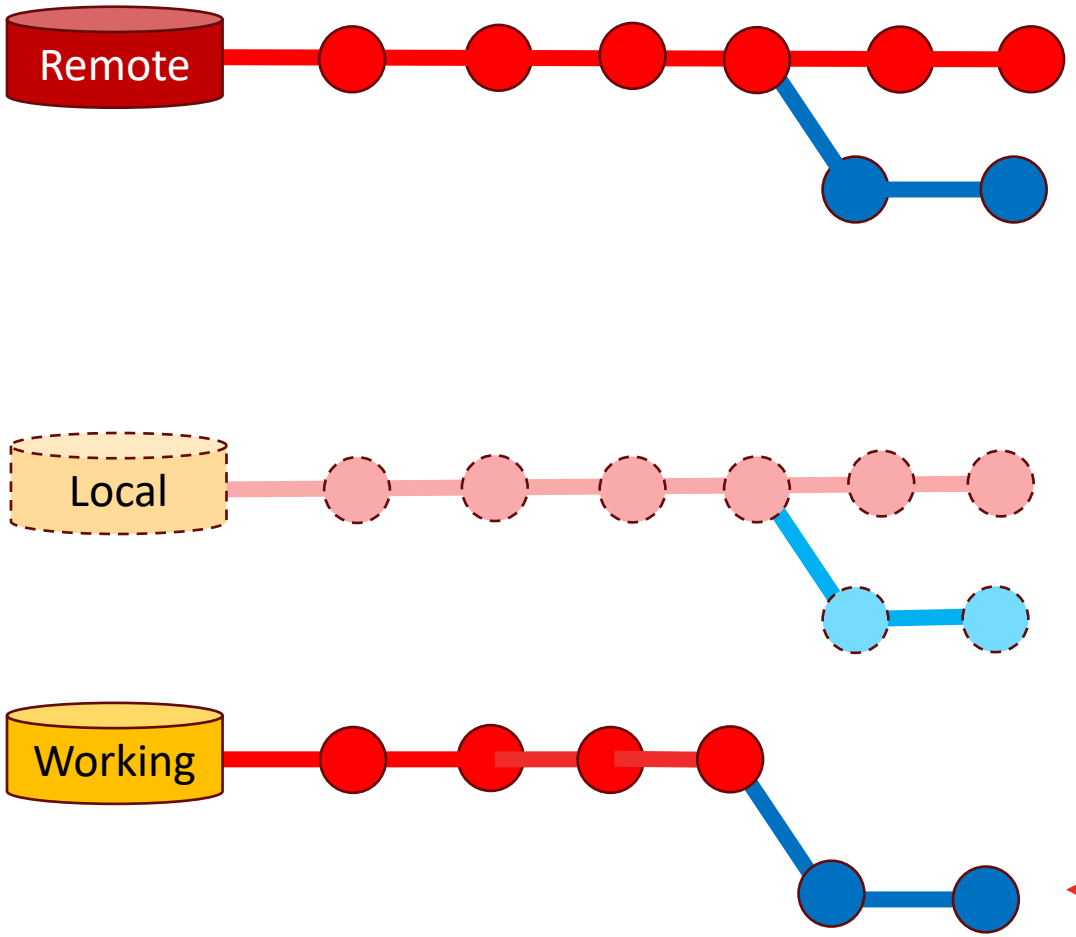


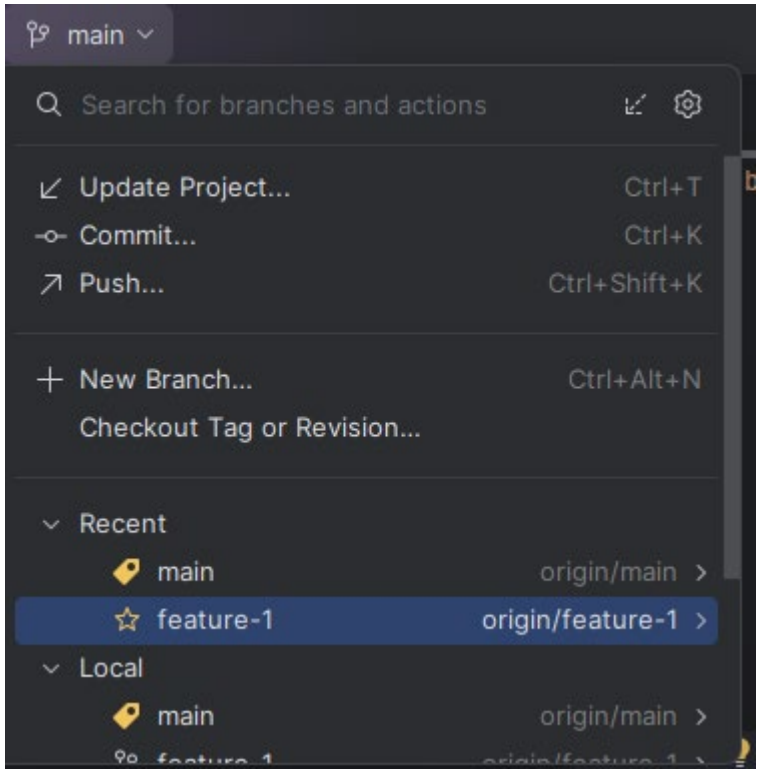


Which updates the “local” as well as the working directories

Using “pull” to update project (Project Update in Intellij)

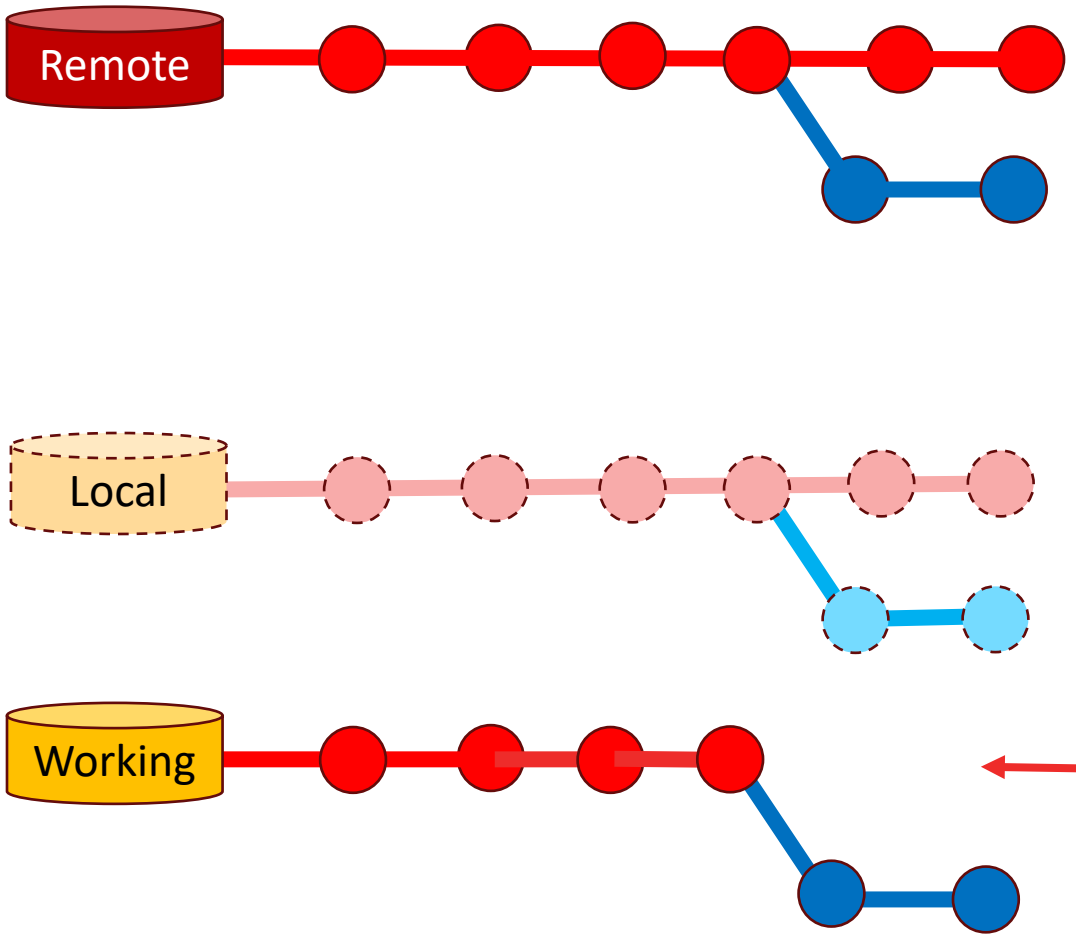
Workflow: Merge -2-





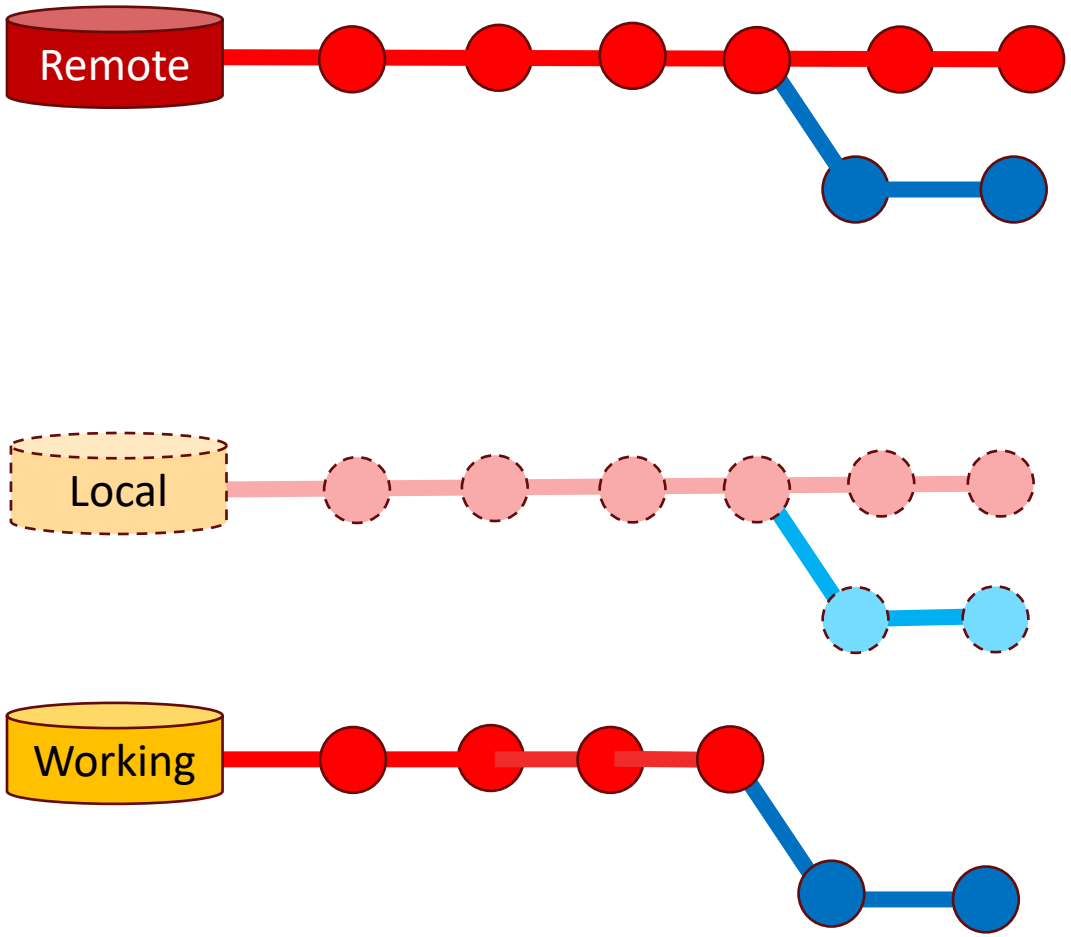
Barry then does a “checkout” to his own branch

Workflow: Merge -2-



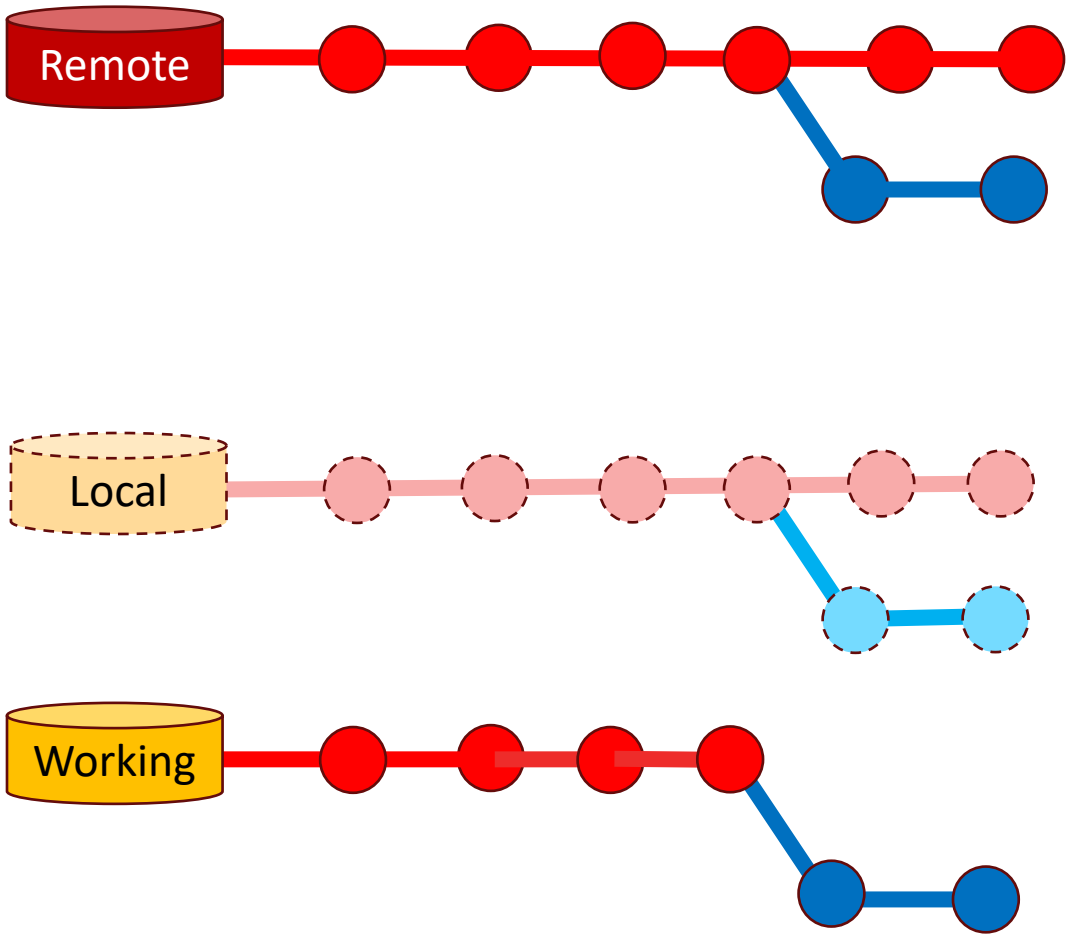
Where did the changes he pulled from remote go?

Workflow: Merge -2-



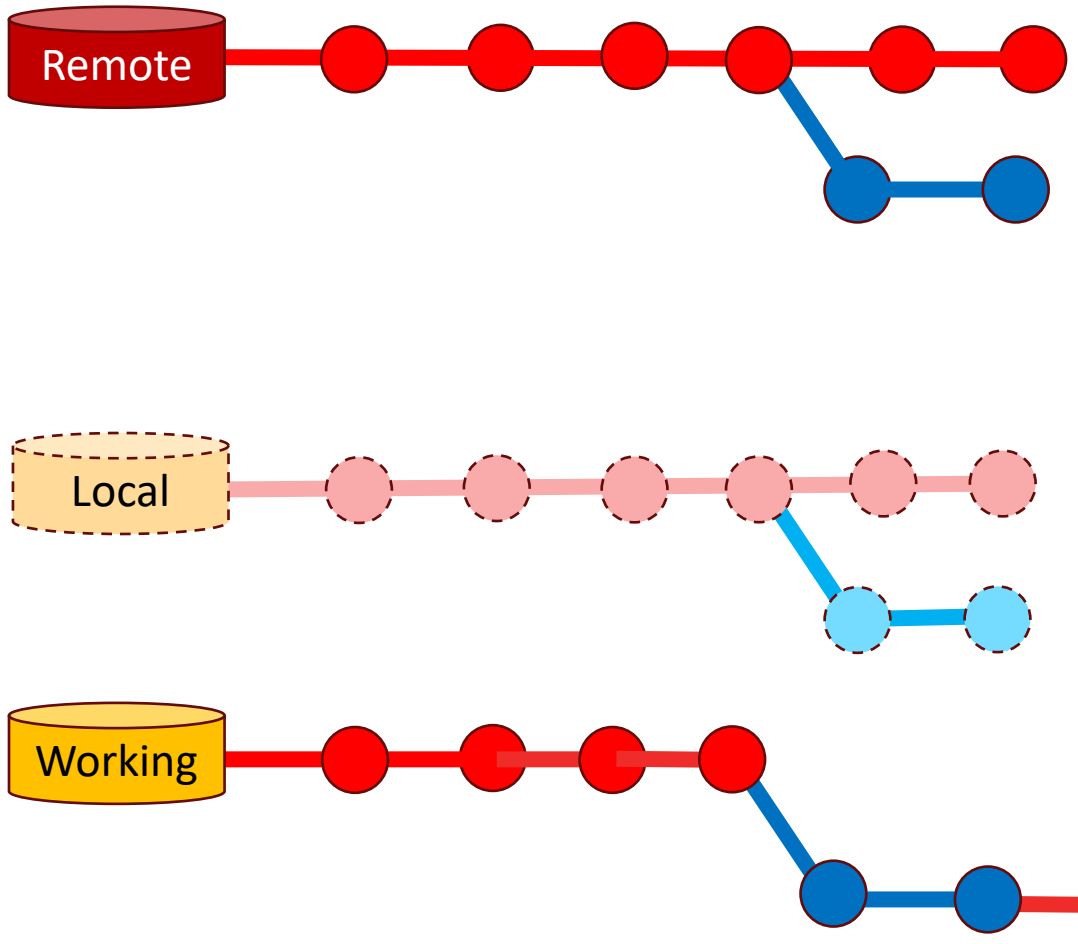
They are still here, working directory only shows the currently active branch.

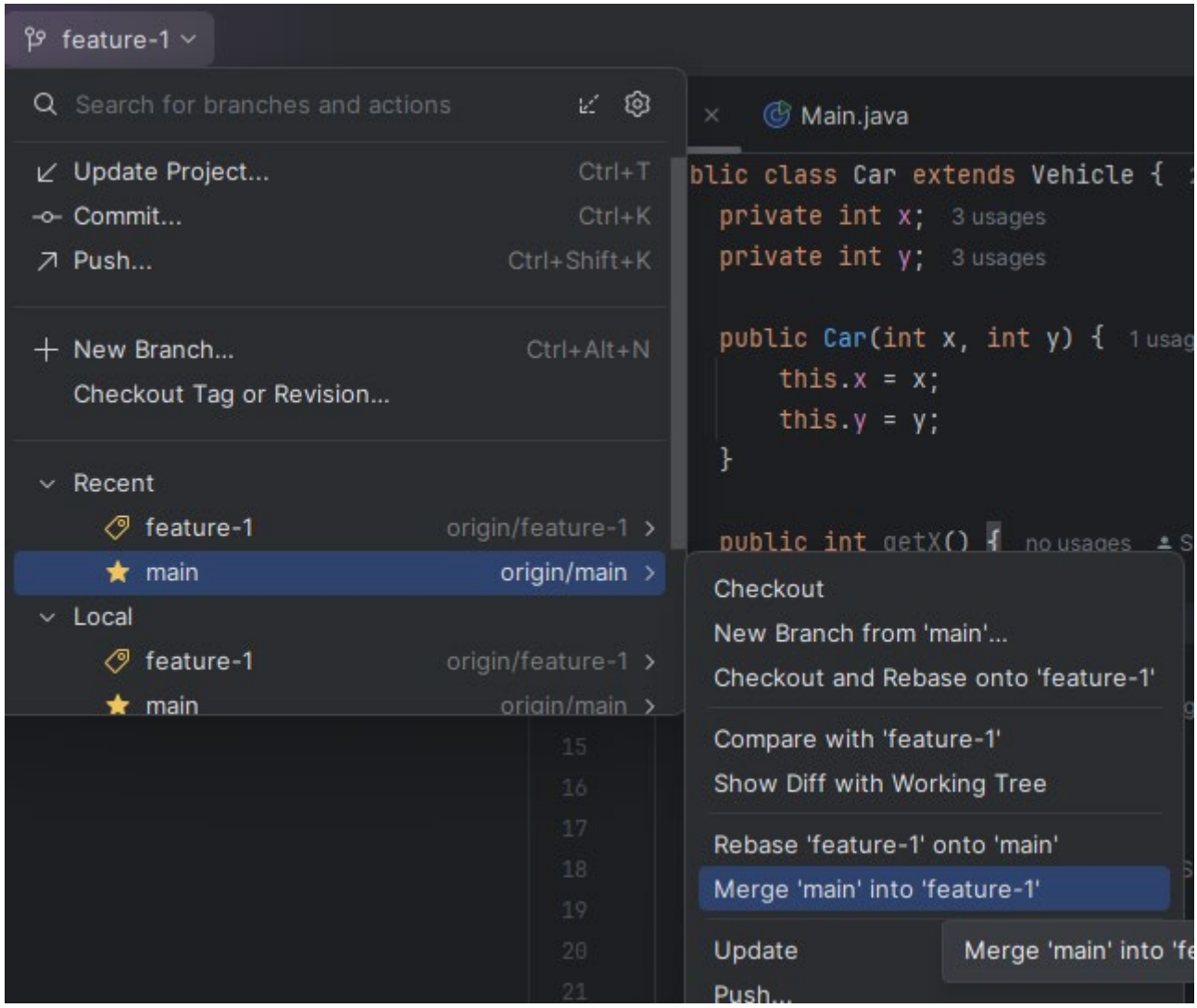
Workflow: Merge -3-



Now Barry can merge the changes from the main branch to his branch

Workflow: Merge -3-

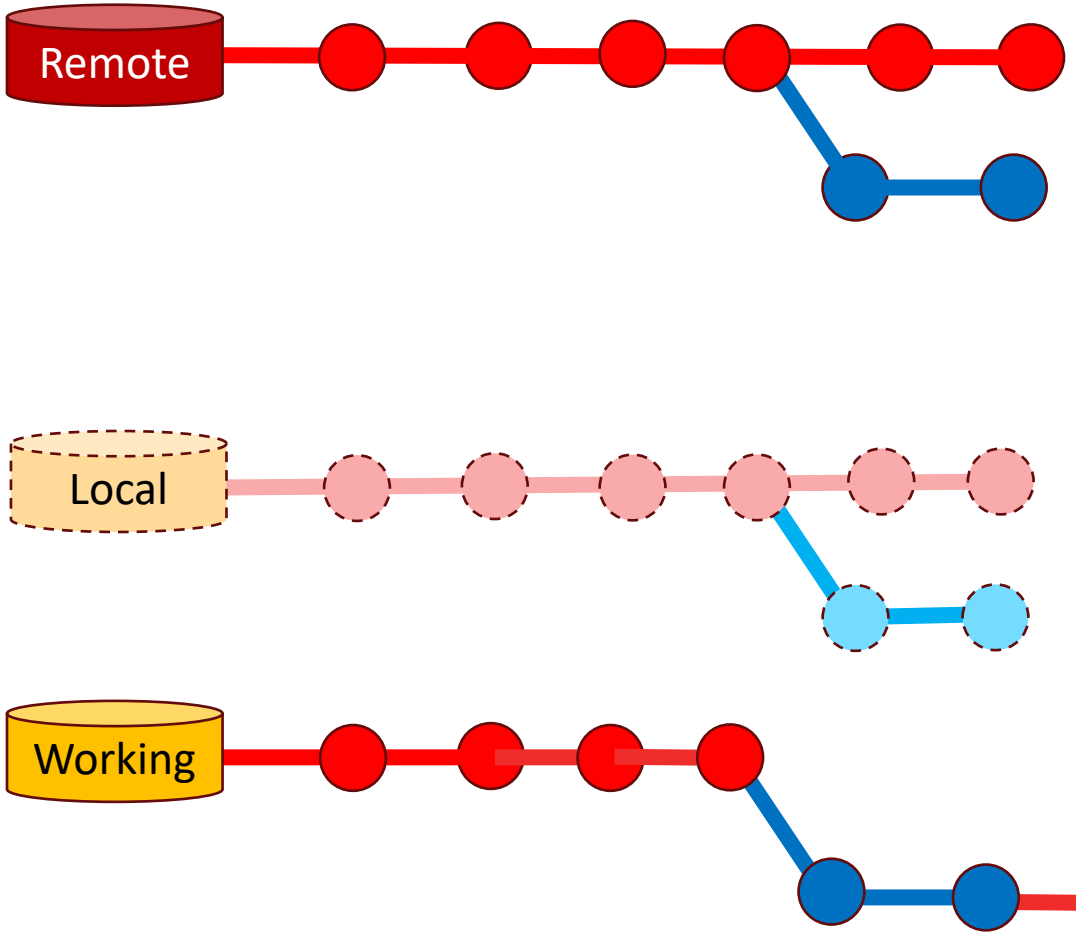


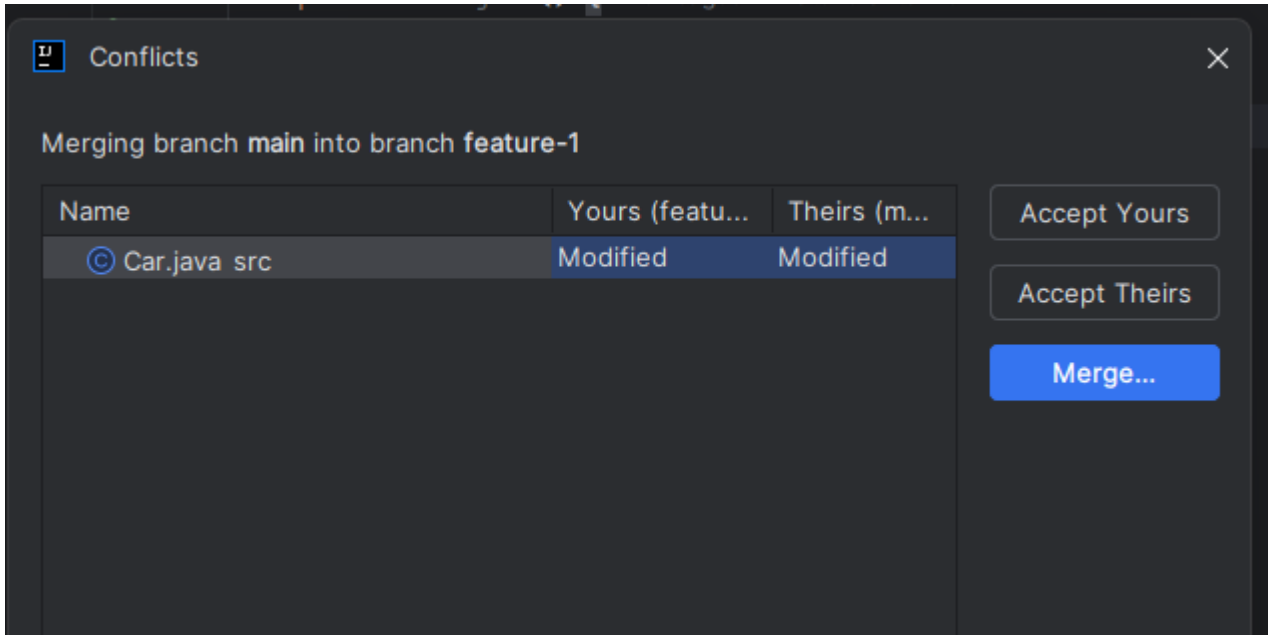


Now Barry can merge the changes from the main branch to his branch

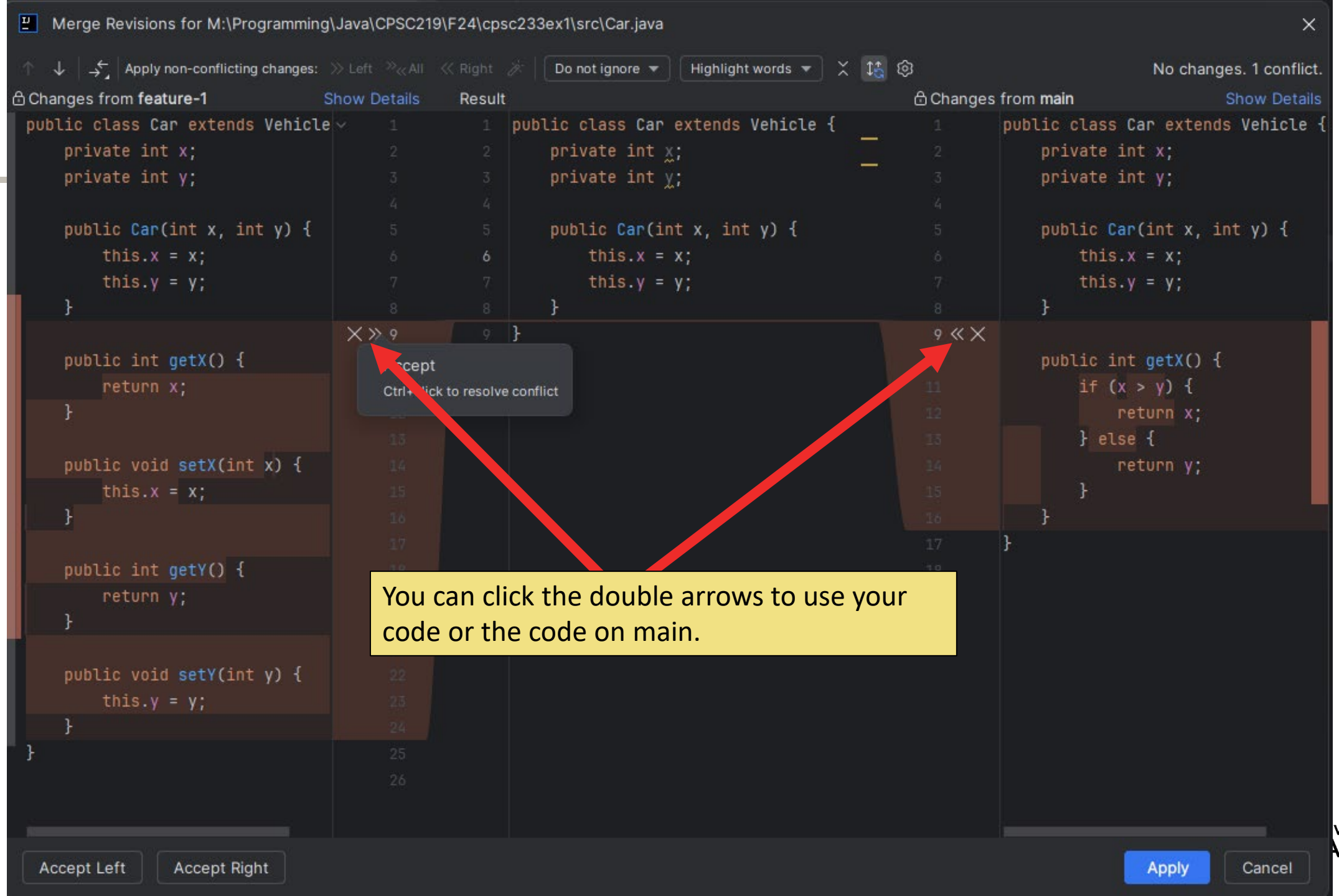
Merge Conflicts

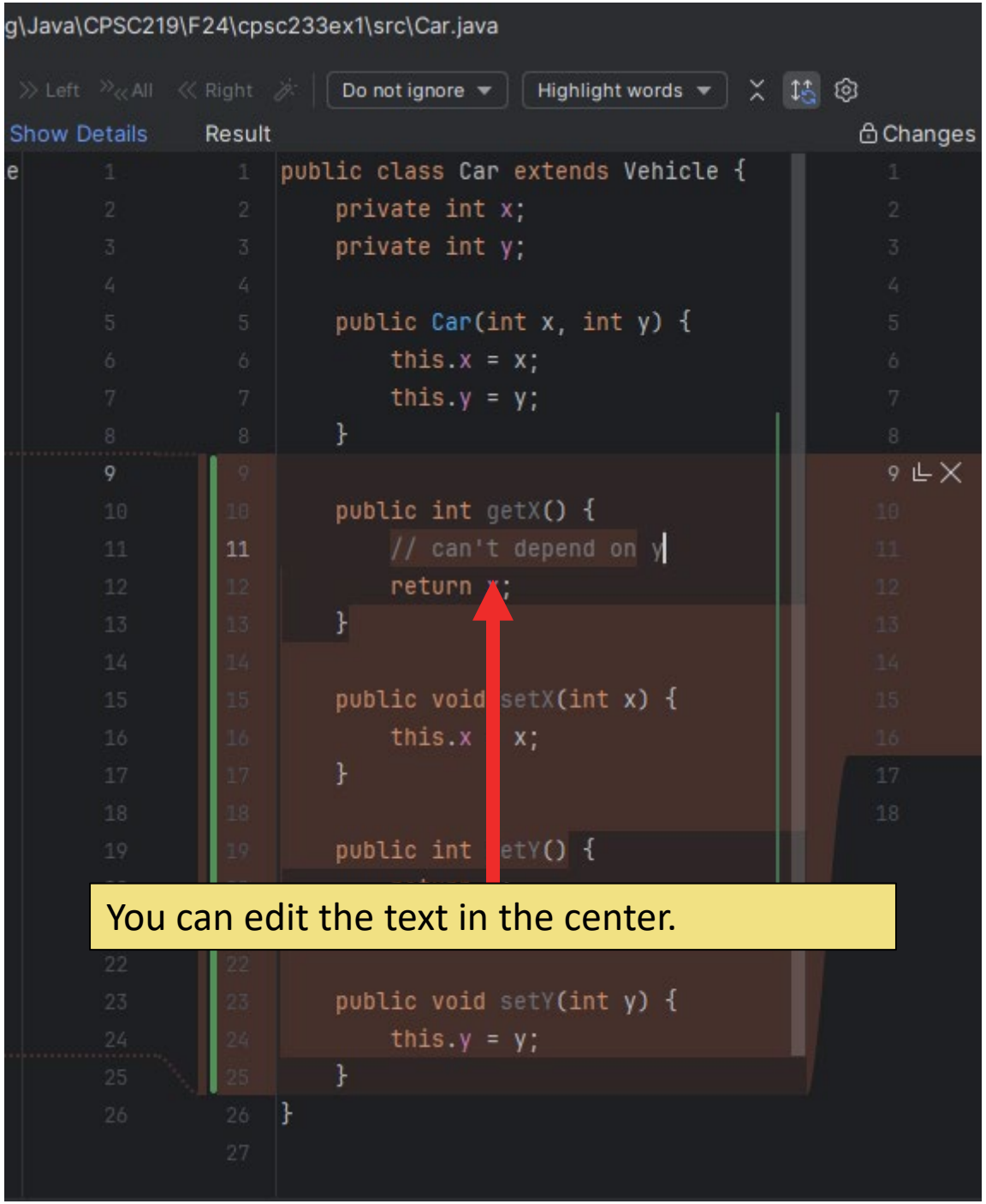
Workflow: Merge -3-



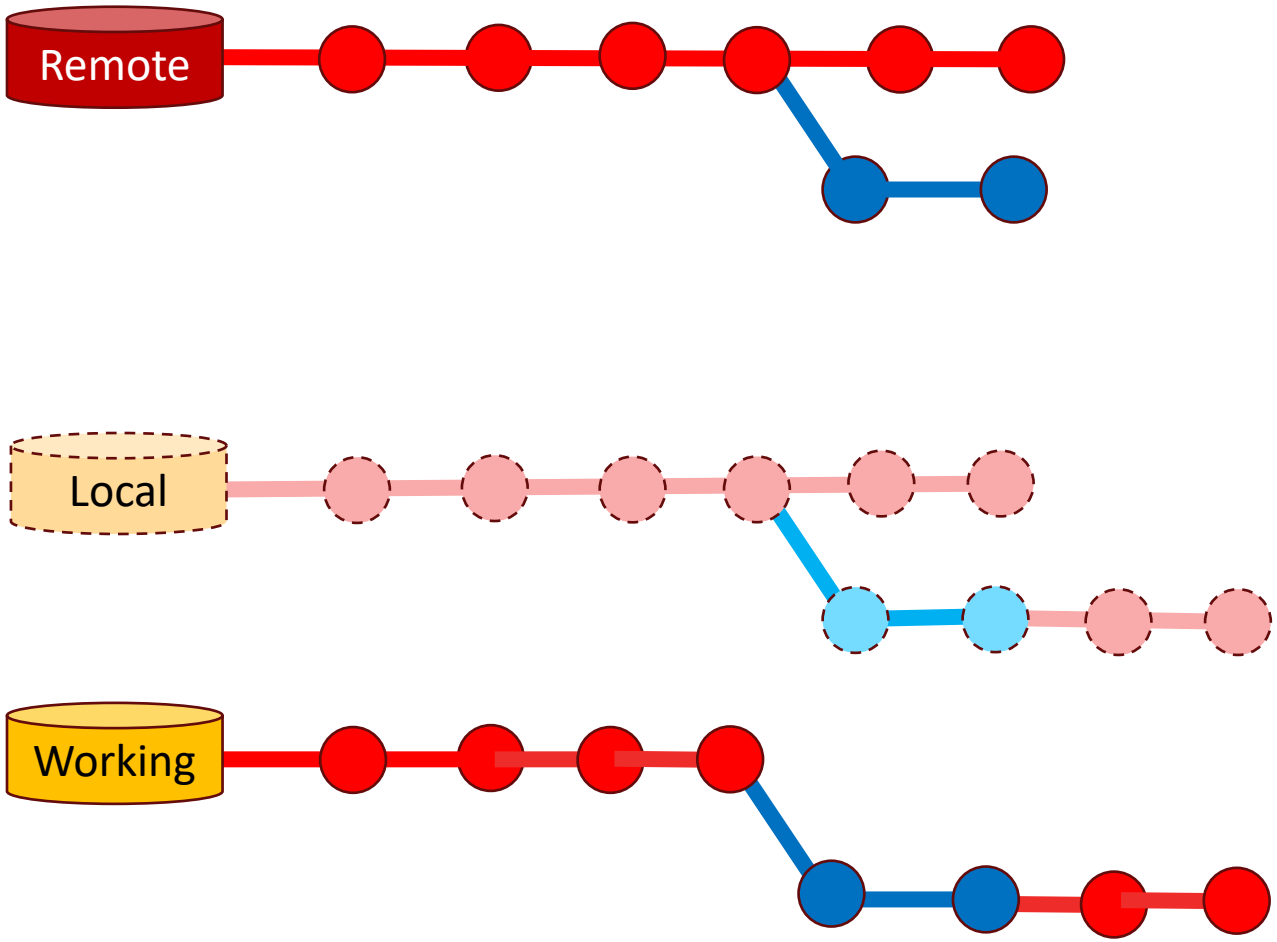


Resolve conflicts if there are any.


Merge Revisions for M:\Programming\Java\CPSC219\F24\cpsc233ex1\src\Car.java
Apply non-conflicting changes:
Left
All
Right
Do not ignore
Highlight words
No changes. 1 conflict.
Changes from feature-1
Show Details
Result
Changes from main
Show Details
public class Car extends Vehicle {
private int x;
private int y;
public Car(int x, int y) {
this.x = x;
this.y = y;
}
public int getX() {
return x;
}
public void setX(int x) {
this.x = x;
}
public int getY() {
return y;
}
public void setY(int y) {
this.y = y;
}
}
Ctrl+ lick to resolve conflict
public int getX() {
if (x > y) {
return x;
} else {
return y;
}
}
You can click the double arrows to use your code or the code on main.
Accept Left
Accept Right
Apply
Cancel

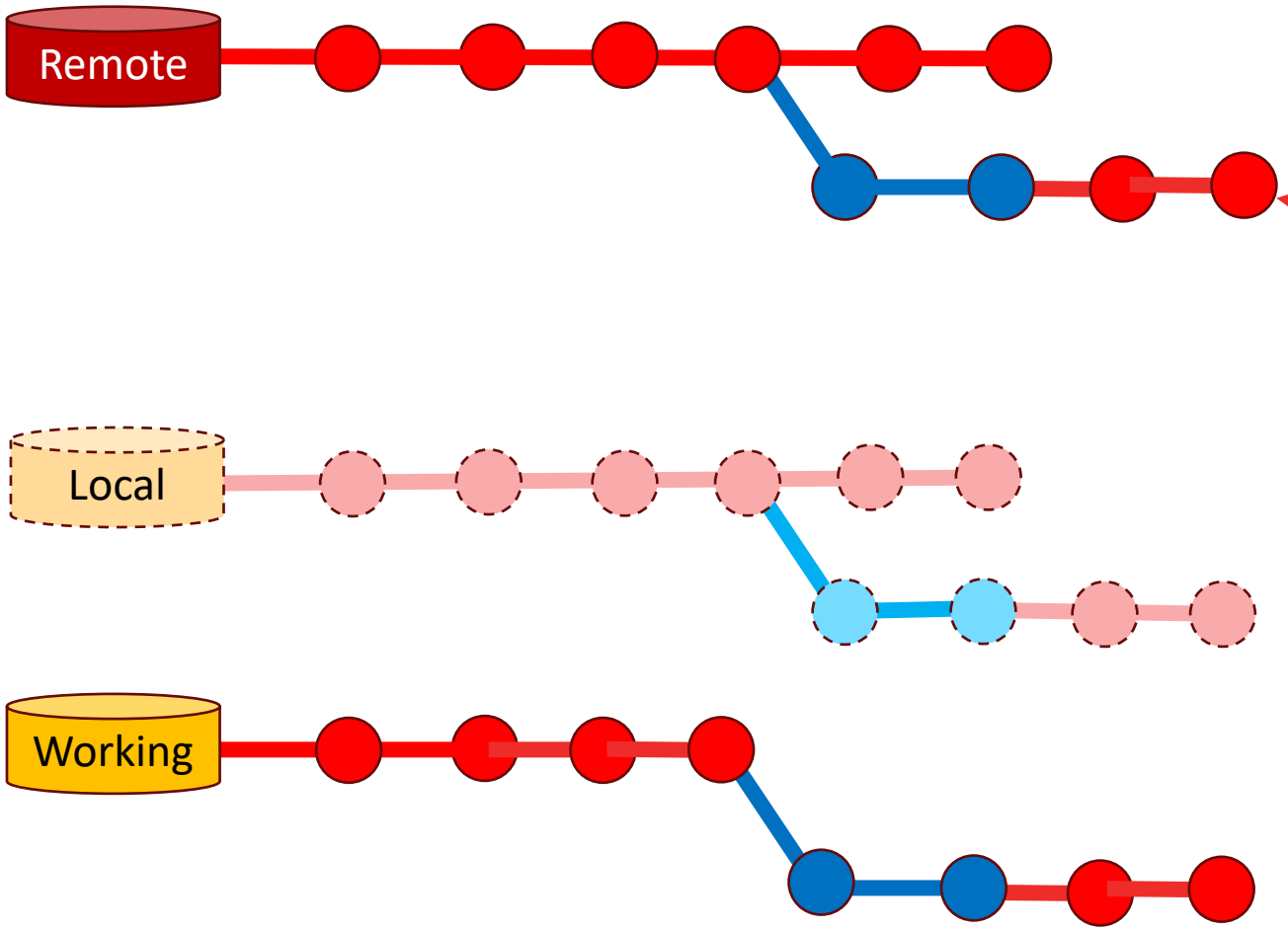


Workflow: Merge -3-

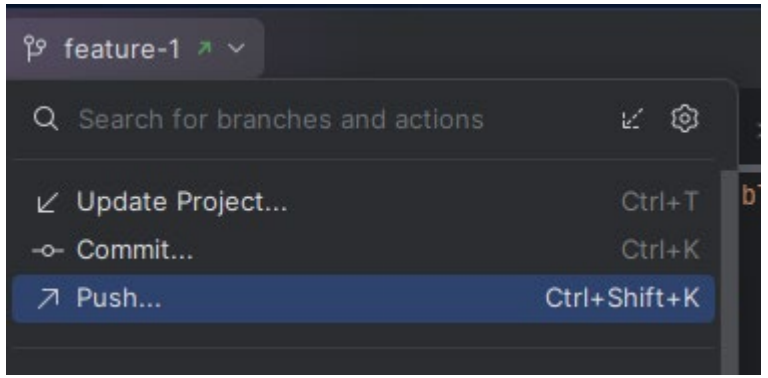


Commit changes (intellij does this automatically for you)

Workflow: Merge -3-

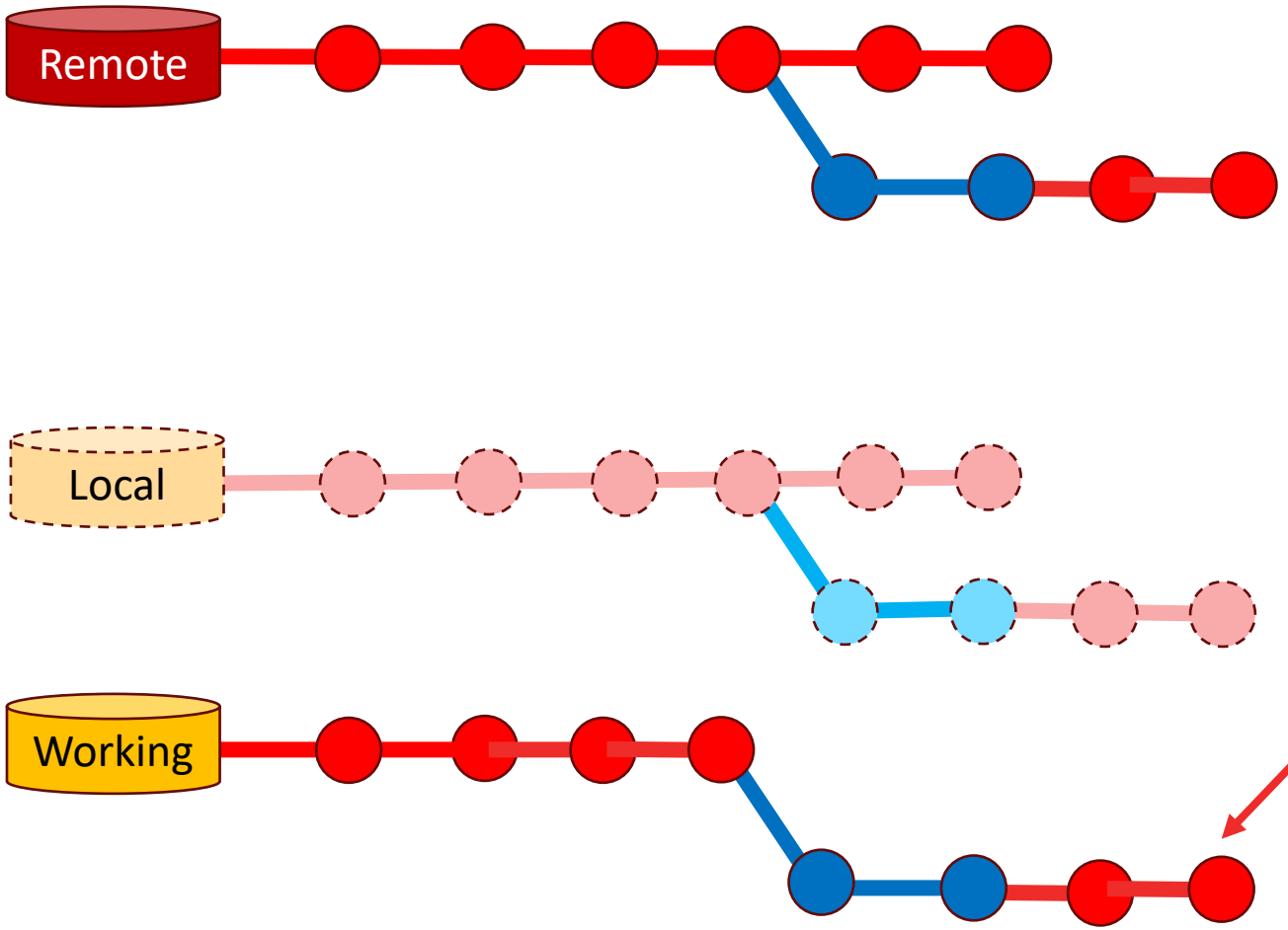


And push those changes to the world



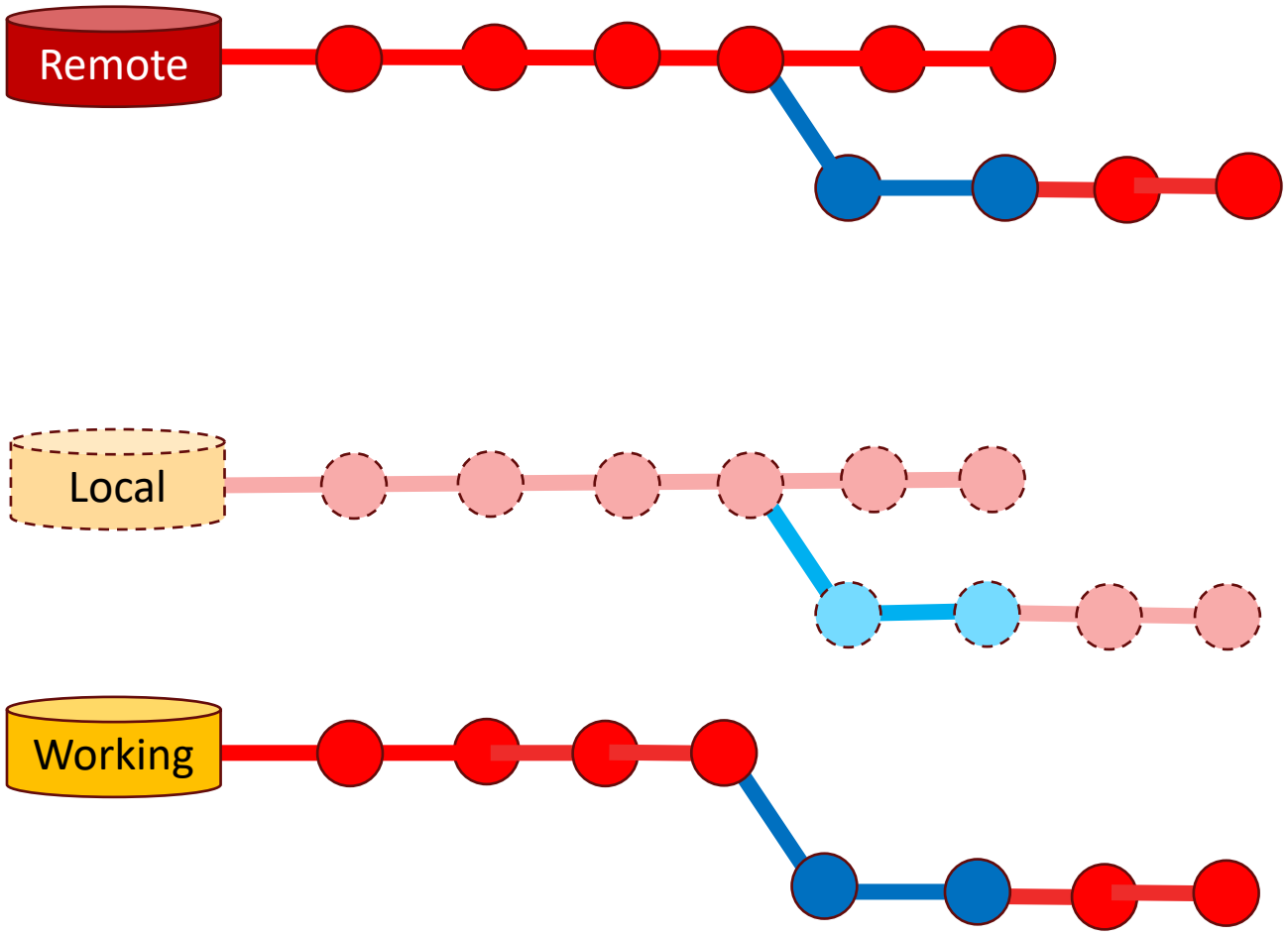
Pull requests

Workflow: Merge -4-

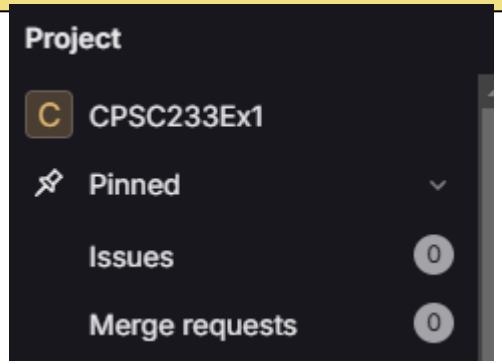


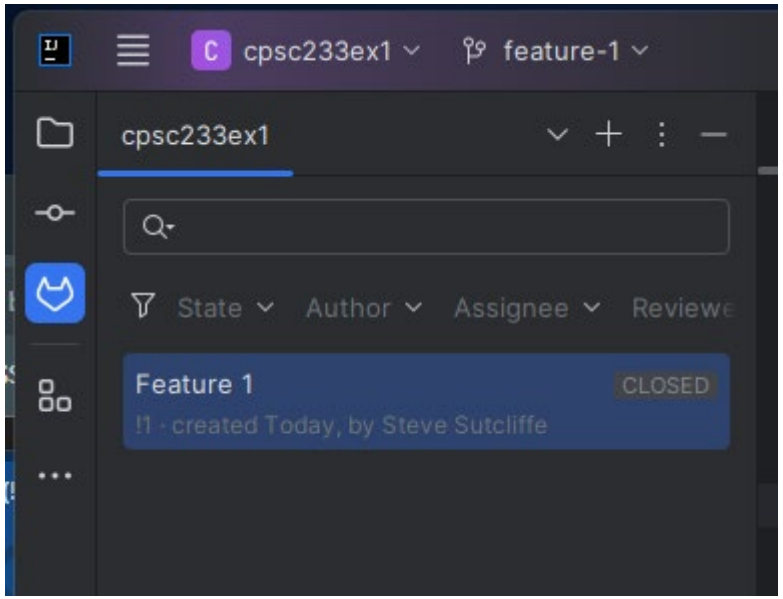
Once Barry is happy with the result, and wants to make his changes permanent

Workflow: Merge -4-



Barry will make a Pull Request (PR), called a Merge Request in Intellij, using the GitLab "Merge Requests" tab, or doing a Merge Request in IntelliJ

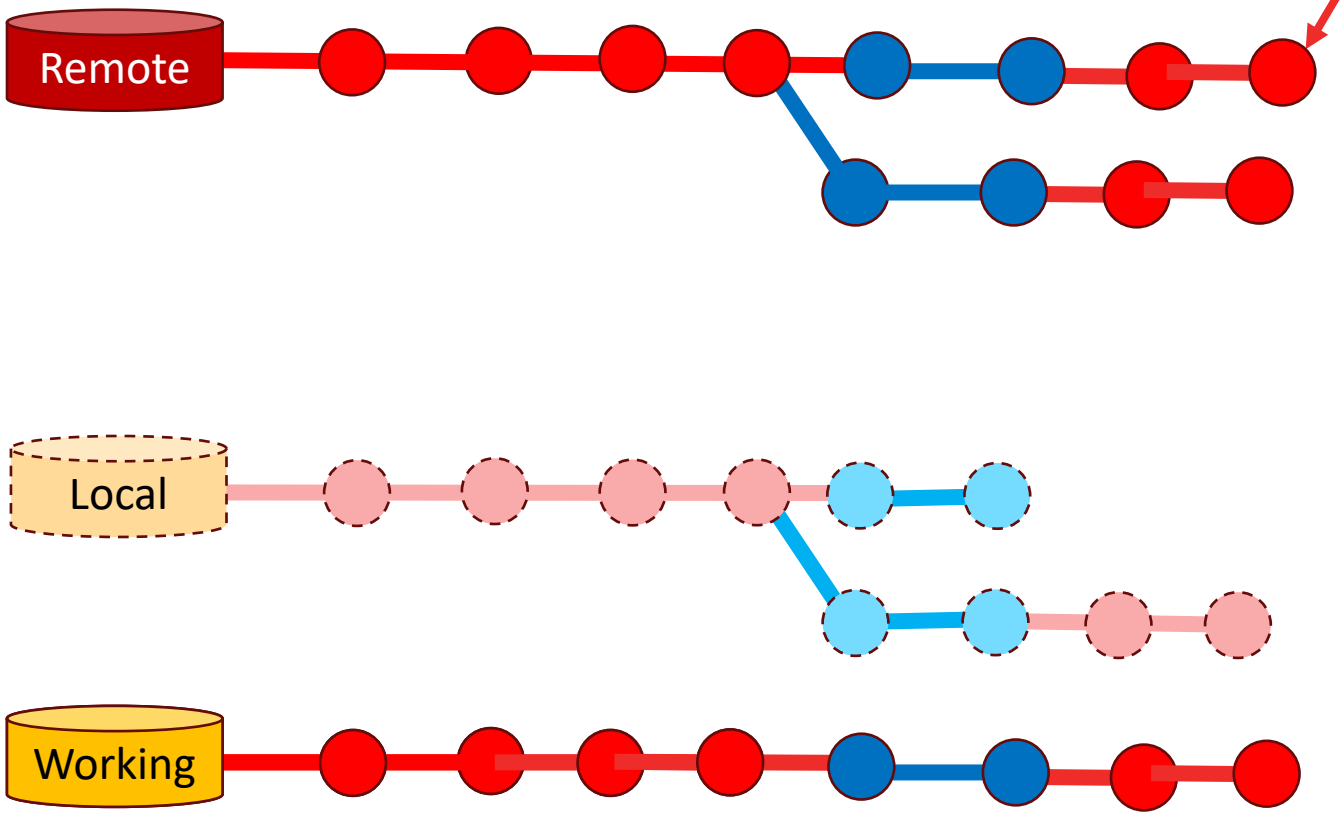


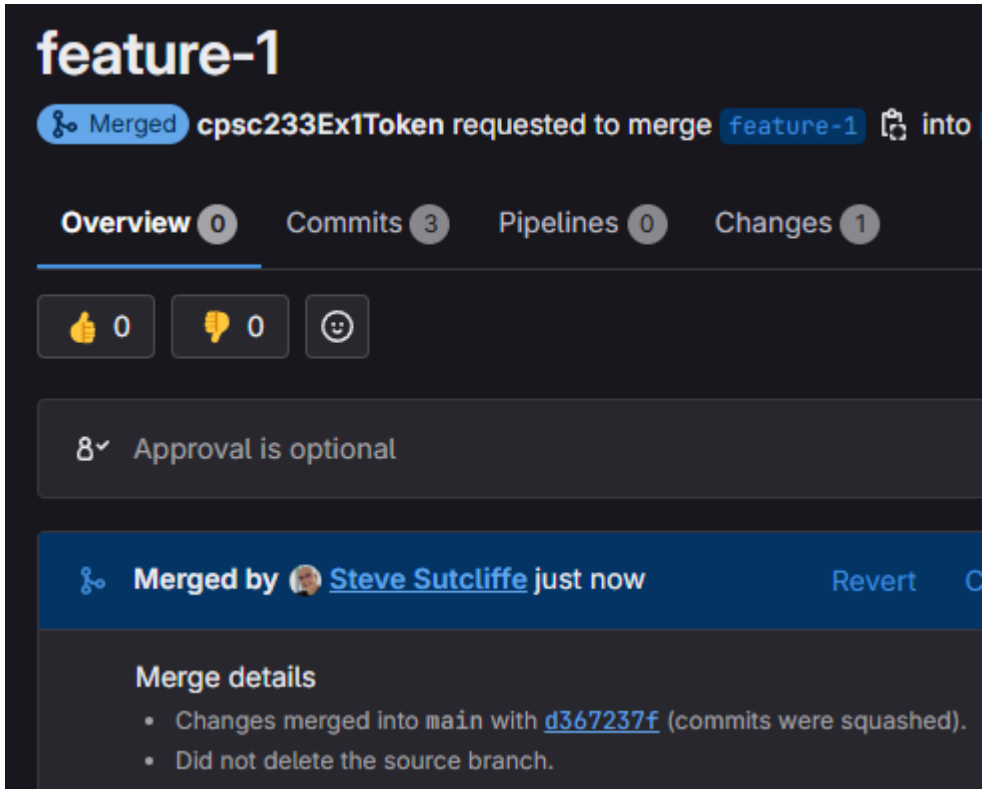


UNIVERSITY OF CALGARY

Workflow: Merge -4-

Then one member of Barry's team will Approve the PR to make the change permanent in remote





UNIVERSITY OF CALGARY

Onward to … System.

Jonathan Hudson
jwhudson@ucalgary.ca
https://pages.cpsc.ucalgary.ca/~jwhudson/

UNIVERSITY OF CALGARY