Article
Nicole Aaron · Dec 21, 2018 5m read

Source control with a shared development instance

The goal of this article is to give users of InterSystems products background on source control in general, and how it can be used with Atelier, an Eclipse plugin. Specifically, I'll be discussing what to do when all of your developers share a single dev instance of an InterSystems product where they can edit and compile code. For more information about the implementation details, and sample workflow, see Michelle Stolwyk's DC post giving a Perforce example. Once you have a better understanding of the concepts and have some ideas of how you might want to implement this on your own system, we suggest reaching out to your InterSystems Sales team to help tailor the setup to your organization ' s needs.

Atelier and source control

Atelier can make using a version control system much easier than it once was with Studio. That is because a third-party source control system depends on files on disk for each of your classes, routines, etc. When you use Studio, those files are all stored in a CACHE.DAT, which source control in general has no concept of. While you work in Atelier you are copying files into a local project for editing, which creates a run-of-the-mill text file version of your code – that is something that a source control system can work with just fine.

However, one complication with using a client-side source control system (the type available via Eclipse plugins) comes when multiple developers share a single development server. This opens the door for developers to overwrite each other ' s code without ever knowing it. You can avoid that by thoughtfully using certain tooling in your source control system, which I'll touch on later in this article. Additional details can be found in Michelle's article.

Some background on source control

In general, there are two different types of source control systems: centralized and distributed.

With a distributed source control system, each developer has their own local copy of the code base and code change history. This allows each developer to make changes to their local code base independently, online or offline. Examples of distributed source control systems include Git and Mercurial.

With a centralized source control system, there is one central record of your code base. Developers check changes into that central code base. The central repository is always the source of truth. Examples of centralized source control systems include Perforce and Subversion (SVN).

If you are using a centralized source control system, there are options for exclusive locking of files. These options are the key to preventing code from being overwritten on a shared development instance.

What ' s the problem with developers making changes on a single dev instance?

Let ' s say you have two developers, Martha and Bob, who share a development InterSystems IRIS server. They both copy the same class into their individual Atelier projects, right around the same time.

A few minutes later Martha makes a change to that class and compiles it on the server. The class compiles successfully, so she ' s good to go and heads off on her lunch break.

Bob had copied the class into his Atelier project before Martha implemented her change. He makes his own changes and compiles on the server. The class compiles successfully, so he thinks he ' s good to go too.

What are the possible problems here?

- Bob overwrites Martha ' s changes without ever knowing they were there, or
- Bob sees the Atelier Conflict Resolution pop-up, indicating that his local copy of the class is different than the server ' s version. He sees Martha ' s change on the server version of the class, but he doesn ' t know who made the change, when it was made, or why it was implemented. He asks around trying to find out who made the change but Martha ' s off to lunch so he doesn ' t have any luck.

What good does source control do me?

If you use a centralized source control system, you have the option to enable exclusive locking on files within the central code base. That way only one user at a time can open a given file for editing. When a developer opens a file with exclusive locking enabled, they will be the only one able to make changes, until they check the file back in therefore releasing the lock.

If a developer must check out a file in order to make changes, and they must take out an exclusive lock to do that, then that leaves you with only one developer at a time who can change and compile code on the server. This type of locking system should only need to occur on your development system, where the code is volatile.

A key piece to keeping this system working is that you must have buy-in from all of your developers. Everyone needs to be using source control, and no one can make changes outside of source control.

Special considerations

This system should not be used in conjunction with server-side source control hooks (Studio hooks). These do not play nicely with a client-side source control model.

If you are thinking about using an exclusive locking model like that described here and in Michelle's article, consider the implications of exclusive locking for your development workflow. Your team needs to agree that it will not be a problem that at any given time, only one developer can have a file checked out and be making changes.

Your team ' s scale matters. These articles were written for a general scenario where we are picturing relatively large development teams. If you are a shop with two developers, and those developers typically work on distinct files, then you may not have anything to worry about here.

Where do I go from here?

Michelle's DC post with a Perforce example gives many of the details left out in this article.

We ' d encourage you to do some general source control research if it is something you are not familiar with. I touch on a few big topics in this article, but there is a lot more information out there when you are considering which source control system is best for you.

Your InterSystems Sales team is a great resource for design-level discussions, such as picking a source control system and implementing that for your development teams.

The InterSystems Support department is available 24/7 to help with specific questions about Atelier and other InterSystems products.

The Developer Community is another great tool. There may be other organizations who have already considered some of those things you ' re wondering about.

#Change Management #Development Environment #Perforce #SVN

Source URL:https://community.intersystems.com/post/source-control-shared-development-instance