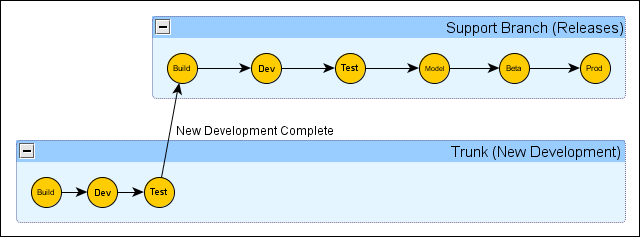
# Deploying Patient Access

Jonathan Lent

# Overview

## How Many Environments Exist?

Patient Access follows a simple, two-branch model of deployment. New development is checked into the Trunk until *all* new development is complete for a particular release. At this point, the code is branched (copied, more or less) from the Trunk to the Support branch. The only work done on code in the Support branch is the fixing of any bugs that might keep the new release from functioning properly when it is deployed to production.



### Environments in the Trunk

There are three environments in the Trunk: Build, Dev, and Test. The Build environment exists only to support the build server and unit tests and is never openly used by actual people. The Dev environment is the first server environment on which new code is run and can be considered alpha-quality. Features in this environment have not undergone any formal testing by the QA team. The Test environment is used by the QA team to perform initial acceptance testing on new features. Since these features have some QA work behind them, the builds in this environment can be considered of beta-quality. Once all features for a release have been completed and initial acceptance testing is complete, the code is branched and the environments in the Trunk are ready to use for the next new release.

### Environments in the Support Branch

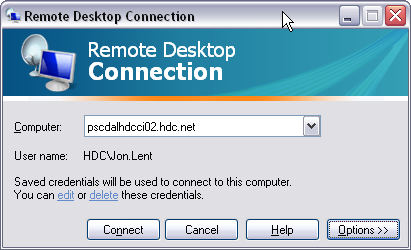
There are six environments in the Support branch: Build, Dev, Test, Model, Beta, and Prod. The Build environment serves the same function as its equivalent in the Trunk. The Dev and Test environments in Support are, frankly, rarely used and one should not be too surprised to see them phased out in the near future. The Model office environment is used by the QA team to perform final regression testing on a build before it is released into a production environment. Once the QA team has deemed a build as “releasable”, that build is deployed to the Beta environment. Beta is actually a production environment with a limited number of users and it typically leads the full production deployment by a week. If no major issues appear in the Beta environment, the build is deployed to the Prod environment where the full user community can access it.

# Running the Script

The act of deploying a build to a particular environment is handed by a single Windows PowerShell script located on the Patient Access build server, **pscdalhdcci02.hdc.net**. To access and run this script, you will need to be a member of the Patient Access development, support, or build team. The script is designed to hide most of the complexity of the deployment process, so it reduces the effort of pushing out a build to two steps.

## Step One: Remote into the Deployment Server

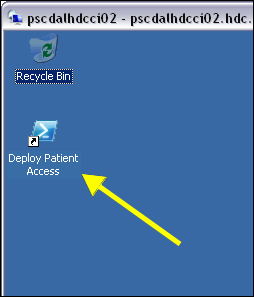
As mentioned above, the deployment server for Patient Access is pscdalhdcci02.hdc.net. You will need to use the remote desktop connection mechanism on your local workstation to connect to this machine so you can run the script:



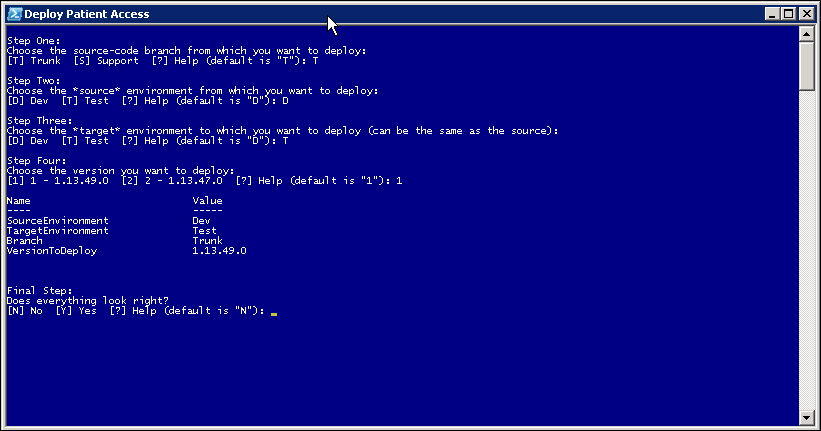
Obviously, you will not be using HDC\Jon.Lent as your user name as the picture shows. Just use your own HDC credentials.

## Step Two: Double-Click the Deployment Icon

Once you have created a remote session, you should notice the “Deploy Patient Access” icon on the desktop:



Tostart the deployment process you just need to double-click this icon. When you do, a PowerShell console window will open and ask you four questions:



The first question (Step One) asks from which branch you want to deploy. The chart at the beginning of this document will prove helpful in determining your answer to this question, but a good rule of thumb is “Trunk” if you are a developer or build team member and “Support” if you are on the operations team.

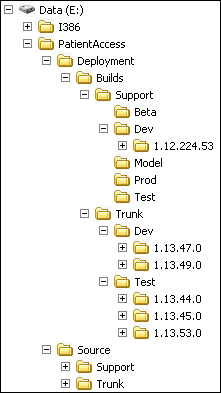
The second question (Step Two) asks you which environment should be used as the source environment. This is the environment that contains the build you want to promote to a new environment. So, if you want to promote the latest support build from the Test environment to the Model environment, you would choose “T” for this value.

The third question (Step Three) asks you which environment will be the target of the deployment. If this value is the same as the environment chosen in Step Two, the deploy script will re-deploy the build to that environment, otherwise it will promote the build from the environment in Step Two to the environment chosen in Step Three.

Before asking question four, the script will scan the build drop folder on the deployment server to see which builds are available for deployment from the source environment. It will then present the list of builds (newest first) to you in Step Four. Choose the number of the build you wish to deploy.

The final step is confirmation of your choices. The script will echo your choices back to you and ask you to confirm the values. If everything looks okay, answer “Y” and the script will start the actual deployment process. If there is a mistake in your choices, just enter “N” and the deployment will be aborted.

The act of deploying the build will generate a LOT of output to the console. This will include information about re-configuring the build for the target deployment, copying the build to the target servers, and updating the application database to the latest version. When the deployment is complete, you can go to the file system on the deployment server and see that the build you selected has been moved to the target environment folder (assume in this case that you asked the script to move build 1.13.53.0 from Dev to Test on Trunk):



**NOTE:** The deployment only keeps the three latest builds for any environment, so don’t be surprised if one or more of the older builds vanish after a successful deployment.

In case you need to re-visit the output of the deployment, you can find the deployment log inside the build folder after a deployment:

