# Workshop HQRM module creation

* Setup Git if attendees don’t have it already
* Setup VS Code if attendees don’t have it installed already (including PS Extension)
* Go over repo. Show x resources, show HQRM resources
* Go over docs. Explain normally one would have to add an issue, etc. Go through guidelines. It’s a lot! Make special not of Style guidline

Start simple

* Create a GitHub repo and clone locally
* Add additional branch locally for code review
* Open the repo in vs code (as Admin, needed for tests and resourcedesigner)
* Install xDscResourceDesigner (writing MOF based as most resources are MOF + compat v4)
* Start writing a simple resource using xDscResourceDesigner (LMHost? it’s not the purpose making the resource a complex one)
* Provide the creation script
* Go through creation result files. Tell about Get Set Test, Singleton, etc.
* Let attendees figure out how to configure LMHost lookup first and have them write own resource?
* Make sure it works. If attendees don’t get it working, provide the code

Testing + Examples + Docs

* Add additional file and folder structure based on online docs  
  <https://github.com/PowerShell/DscResources/blob/master/CONTRIBUTING.md#submitting-a-new-resource-module>
* Add unit test based on test template <https://github.com/PowerShell/DscResources/blob/master/Tests.Template/unit_template.ps1>
* Fill in required fields and disable all tests examples for now (comment out)
* Setup Test task runner and test once so meta testers are downloaded
* Explore what is tested
* Adjust task runner to not check markdown and examples (will skip nodejs, gulp etc for now) Protect attendees 😊
* Adjust keybinding for task runner to Ctrl+T
* Have attendees fix all issues, if attendees don’t get it correct, provide code
* Write unit test (include mocking, etc)
* Guide attendees on Test-TargetResource test and have them develop additional tests
* When done show completed simple example and discuss.
* Add integration test based on test template   
  <https://github.com/PowerShell/DscResources/blob/master/Tests.Template/integration_template.ps1>   
  <https://github.com/PowerShell/DscResources/blob/master/Tests.Template/integration_config_template.ps1>
* Adjust the templates with simple integration test
* Add example file. Create symboliclink to resource. Enable example testing and show result  
  Will throw locally but Appveyor should work correctly.
* Remove symbolic link and run test again
* Add README and appveyor.yml from DSC resource template

Adjust README and appveyor.yml

* Add gitignore file to exclude DSCResource.Tests folder
* Setup appveyor integration
* Adjust manifest
* Commit -> push and see appveyor in play

Now let’s take a close look at the style guide and see where we need to adjust / update the code (if still time). Add localization files. Function comment based help, helper functions terminating error + tests

* Create review PR