

Version Control

WTF is git?

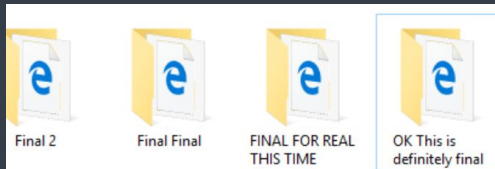
Familiar experience?



I prefer the real version control



I said the *real* version control



Perfection

Comparing changes between

```
Side-by-side viewer | Do not ignore | Highlight words | ?
a22523cac1bc211da207bd8dedb53487bca890b (Read-only) | LF | Local (2a4387aef88dcfaca5208c95a0621ee9e8debb) | 22 differences | CR/LF

def import_module_configs(self):
    config = configparser.ConfigParser()

    # (Import | Freak out over) module config
    for module in self.discover_modules():
        module_config = self.module_path + '/%s/%s.ini' % (module, module)

        try:
            # Attempt to read current module's config file
            self.debug(module_config)
            Path(module_config).resolve()

        except FileNotFoundError:
            # Was unable to read this module, log an error then skip
            self.debug(module + " config file not found!")
            continue

        else:
            # Module config exists, start importing datas
            self.debug(module + " config file found, importing data")
            config.read(module_config)

            try:
                # get module_desc, options, fw_requirements, output_format, version, module_help
                current_module = ModuleDescriptor(
                    module_name=config.get("general", 'module_name'),
                    module_desc=config.get("general", 'module_desc'),
                    version=config.get("general", 'version'),
                    module_help=config.get("general", 'module_help'),
                    # _sections[section] returns as a dictionary
                    options=config._sections['options'],
                    fw_requirements=config._sections['fw_requirements'],
                    output_format=config._sections['output_format']
                )
                self.module_list.append(current_module)

            except configparser.Error:
                self.debug("Error: Unable to import module from file")
            else:
                self.debug("modules loaded: " + str([module.module_name for module in self.module_list]))

# get module_desc, options, fw_requirements, output_format, version, module_help
current_module = ModuleDescriptor(
    module_name=config.get("general", 'module_name'),
    module_desc=config.get("general", 'module_desc'),
    version=config.get("general", 'version'),
    module_help=config.get("general", 'module_help'),
    # _sections[section] returns as a dictionary
    options=config._sections['options'],
    fw_requirements=config._sections['fw_requirements'],
    output_format=config._sections['output_format']
)
self.module_list.append(current_module)

# Prevent ordered dict from duplicating everything
config.sections["general"] = {}
config.sections["options"] = {}
config.sections["fw_requirements"] = {}
config.sections["output_format"] = {}

except configparser.Error:
    self.module_manager.debug("Error: Unable to import module from file", color=Format.color_warn)
else:
    self.update_order(module_name=this_module)

self.module_manager.debug("modules loaded: " + str([module.module_name for module in self.module_list]))

def update_order(self, module_name):
    try:
        module = self.get_module_by_name(module_name)


        if "true" == module.options["enabled"].lower():
            try:
                # If it already exists it needs to be removed first
                self.module_order.remove(module_name)
            except Exception:
                # Easier to ask for forgiveness
                pass

            # Add module to order
            self.module_order.append(module_name)
            self.module_manager.debug("%s added to order" % (module_name), color=Format.color_info)

        else:
            try:
                self.module_order.remove(module_name)
            except Exception:
                pass # probably don't need to remove it
            else:
                self.module_manager.debug("%s removed from order", color=Format.color_success)
```

branches to experiment with new features...

Default branch

 **master** Updated 15 days ago by Joh98



Default

Change default branch

Your branches

pi-testing Updated 24 days ago by Andrew Calder

45 | 0

#18  Merged 

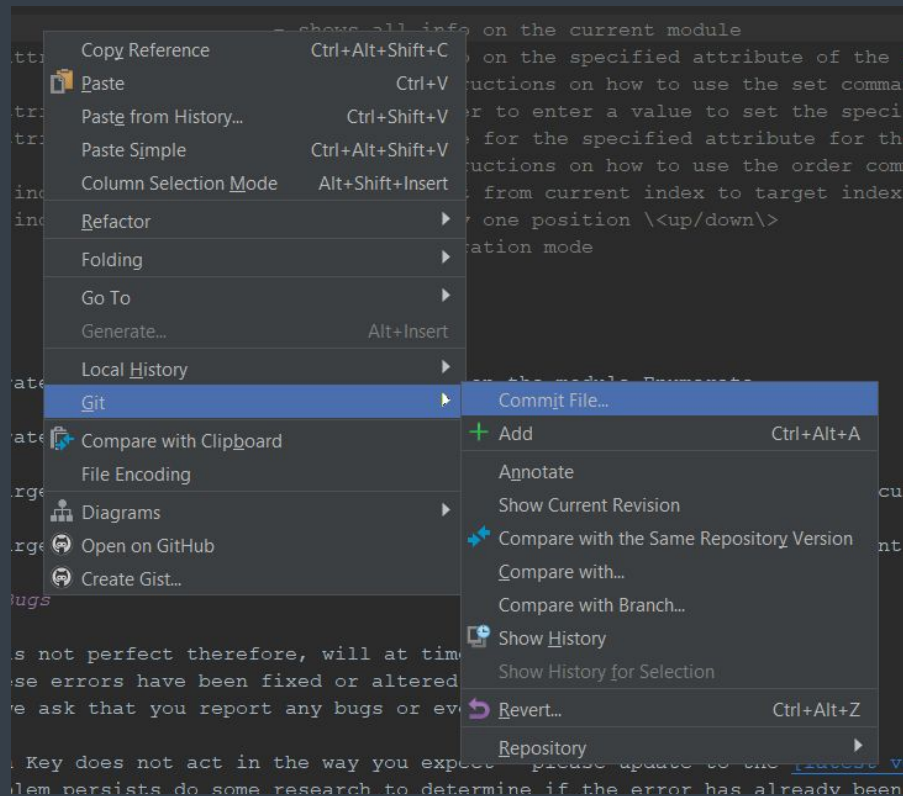
Easily
accessing
current
version
between
devices

**If I Put A Laptop On
A Desk...**



**Does It Become A
Desktop?**

Most IDEs have make it even easier to use



Really easy to get started

```
git init                // create empty git repo in current directory
git init <dir>          // create empty git repo in directory.

git add <dir/file>      // add an individual directory or file to staging
git add .               // add all changed, new and not ignored. Does not add rm actions.
git add -u              // add only changed or removed files/dirs to staging
git add -A              // shortcut for doing 'git add .; git add -u'

git remote add origin <REPO URL> // Sets the new remote (required to push to github repo)
git remote -v           // Verify connection to remote repo

git diff                // show changes not yet added to staging
Git diff --cached       // show changes about to be committed

git commit -m <msg>      // store current version in git with msg. commits need messages.
Git commit -a -m <msg>  // shortcut for 'git add -u; git commit -m <msg>'

git clone <repo>         // clone a repo e.g. 'git clone https://github.com/AbertayProgrammers/' 🐙

git push origin master  // push changes to remote repo
git push                // can be shortened to this once a remote repo has been defined

git help                // for all other commands not covered here
git help <command>      // for help with a specific command
```