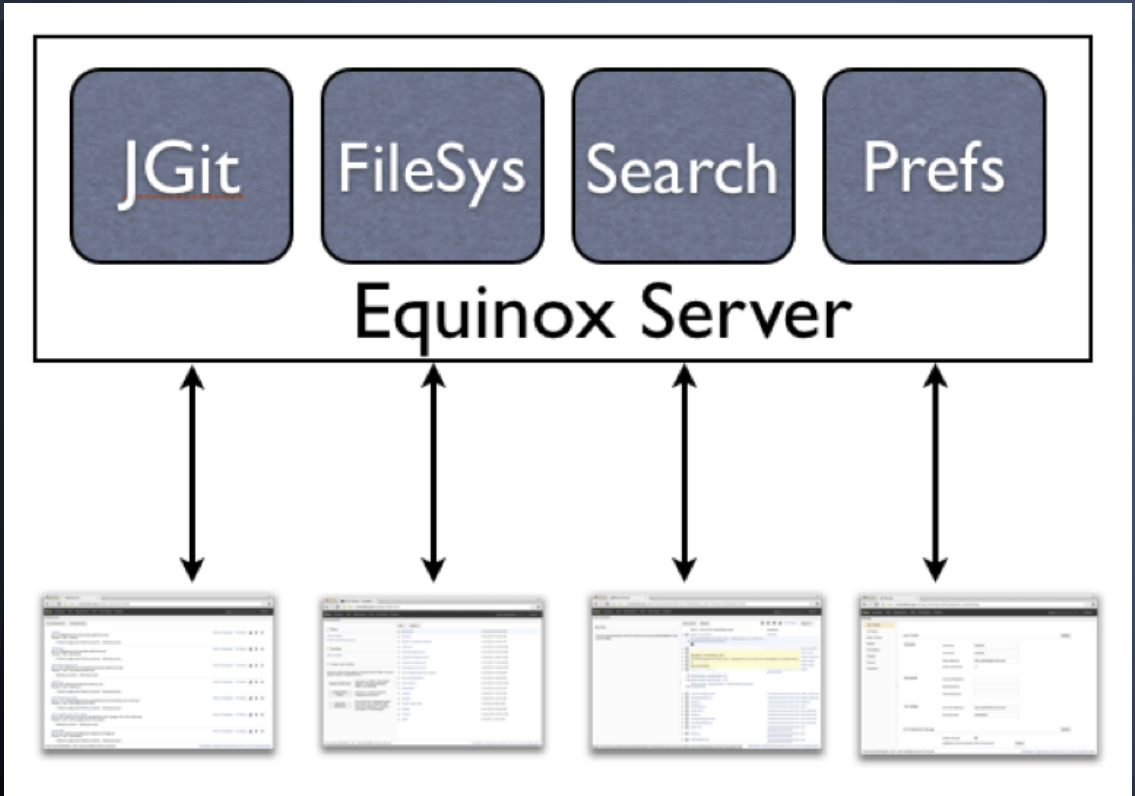
Eclipse-Orion OA Project

Albert Cui & Mei-Vern Then

Our Team

- Paul (mentor)
  - IBM
  - Canadian (eh?)
- Brandon
  - UW
  - Amurrican
- Lef
  - MIT
  - Greek
- Albert & Mei-Vern
  - CU

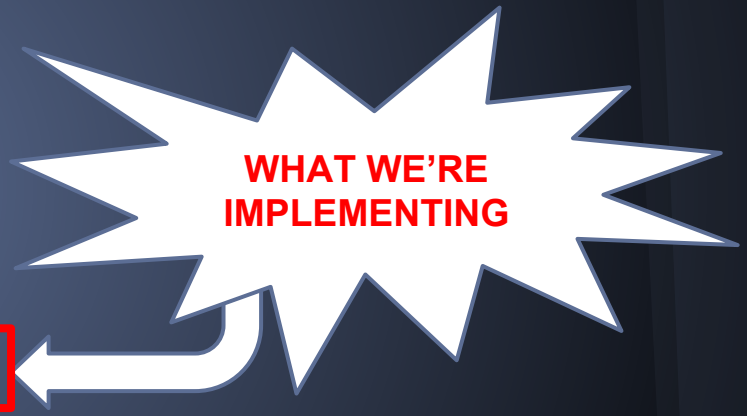
What is Eclipse Orion?



- Cloud IDE
- Two parts:
  - Orion Client
  - Orion Server
- Features:
  - file editing
  - terminal
  - git

Server: Java vs. Node

	Java	Node
Users	multiple	single
Login API (`/login`)	✔	
File operations (`/file`)	✔	✔
Import zip/HTTP/SFTP (`/import`)	✔	
Export zip/SFTP (`/export`)	✔	
Persistent user preferences (`/prefs`)	✔	
Git (`/gitapi`)	✔	
Deploy to Cloud Foundry (`/cfapi`)	✔	
Sites (`/sites`)	✔	
User API (`/users`)	✔	
Integrated system shell (pty)		OS X and Linux only



Goals

- Replicate Java server Git API endpoints on the Node server
  - Client side code has already been written
  - Use NodeGit and the Orion API to complete the task
  - Frontend UI completely taken care of

BEFORE OPEN ACADEMY

Example Code

```
.use(resource(workspaceRoot, {
    GET: function(req, res, next, rest) {
        var query = url.parse(req.url, true).query;
        if (rest === '') {
            console.log("nope");
        } else if (rest.indexOf("clone/workspace/") === 0) {
            clone.getClone(workspaceDir, fileRoot, req, res, next, rest);
        } else if (rest.indexOf("remote/file/") === 0) {
            remotes.getRemotes(workspaceDir, fileRoot, req, res, next, rest);
        } else if (rest.indexOf("remote/") === 0 ) {
            var remote = rest.replace("remote/", "").substring(0, rest.lastIndexOf("/file/")-"/
            if (remote.indexOf("/") === -1) {
                remotes.getRemotesBranches(workspaceDir, fileRoot, req, res, next, rest);
            }
            else {
                remotes.getRemotesBranchDetail(workspaceDir, fileRoot, req, res, next, rest);
            }
        } else if (rest.indexOf("branch/file/")===0) {
            branches.getBranches(workspaceDir, fileRoot, req, res, next, rest);
        } else if (rest.indexOf("branch/")===0) {
            branches.getBranchMetadata(workspaceDir, fileRoot, req, res, next, rest);
        } else if (rest.indexOf("status/file/") === 0) {
            status.getStatus(workspaceDir, fileRoot, req, res, next, rest);
        } else if (rest.indexOf("config/clone/file/") === 0) {
            config.getConfig(workspaceDir, fileRoot, req, res, next, rest);
        } else if (rest.indexOf("config/") === 0) {
```

```
function getRemotes(workspaceDir, fileRoot, req, res, next, rest) {
    var repoPath = rest.replace("remote/file/", "");
    var fileDir = repoPath;
    repoPath = api.join(workspaceDir, repoPath);
    var repo;

    git.Repository.open(repoPath)
    .then(function(r) {
        repo = r;
        return git.Remote.list(r);
    })
    .then(function(remotes){
        var r = [];
        async.each(remotes, function(remote, cb) {
            git.Remote.lookup(repo, remote)
            .then(function(remote){
                var name = remote.name();
                r.push({
                    "CloneLocation": "/gitapi/clone/file/" + fileDir,
                    "IsGerrit": false, // should check
                    "GitUrl": remote.url(),
                    "Name": name,
                    "Location": "/gitapi/remote/" + name + "/file/" + fileDir,
                    "Type": "Remote"
                });
                cb();
            });
        }, function(err) {
            var resp = JSON.stringify({
                "Children": r,
                "Type": "Remote"
            });
            res.statusCode = 200;
            res.setHeader('Content-Type', 'application/json');
            res.setHeader('Content-Length', resp.length);
            res.end(resp);
        });
    })
}
```

1. Parse the route
2. Use nodegit to open the repo and get list of remote names
3. Look up each remote and populate response for each remote
   a. Async / sync issues
4. Send response with correct headers

DEMO TIME!

1. Writing test to see if getting a list of remotes works

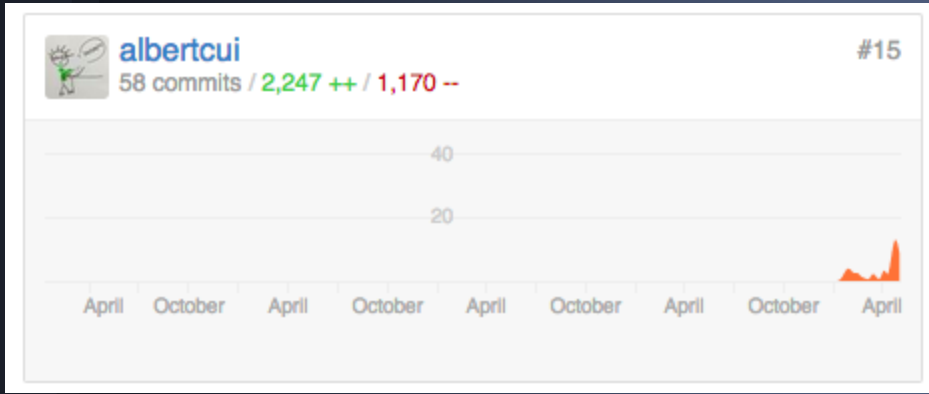
2. Test the response of the route using assertions

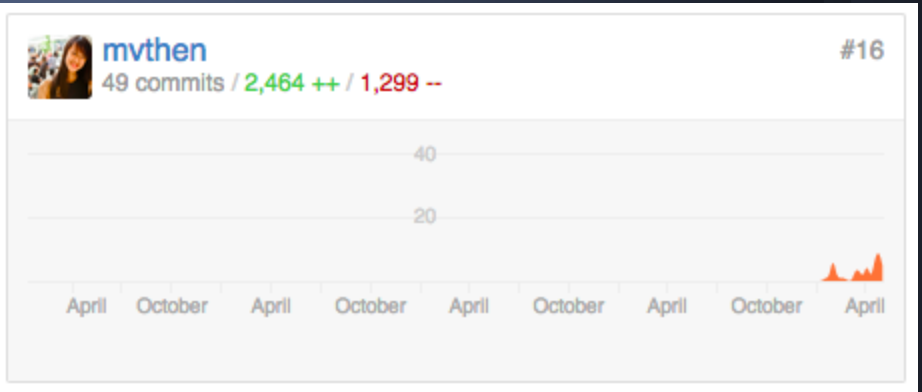
3. Re-check using nodegit

```
describe('Get list of remotes',  function() {
    var numRemotes;

    it('GET remote (getting the list of remotes)', function(finished) {
        app.request()
        .get(CONTEXT_PATH + "/gitapi/remote/file/" + TEST_REPO_NAME)
        .expect(200)
        .end(function(err, res) {
            assert.ifError(err);
            assert.equal(res.body.Children[0].Name, remoteName);
            numRemotes = res.body.Children.length;
            finished();
        })
    })

    it('Check nodegit for list of remotes', function(finished) {
        git.Repository.open(repoPath)
        .then(function(repo) {
            return git.Remote.list(repo);
        })
        .then(function(list) {
            assert.equal(list.length, numRemotes);
            assert(list[0], remoteName);
        })
        .catch(function(err) {
            assert.ifError(err);
        })
        .done(function() {
            finished();
        })
    })
})
```

RUN TESTS!

All the work we've done!





Difficulties

- Poor documentation
  - Both the Orion API and the NodeGit API
- NodeGit isn’t fully featured
  - API also changing between releases
- Orion Node uses old, outdated libraries
- Minimize use of outside dependencies
  - Results in very roundabout solutions
- Communication

Work to be Done

- Rebasing
- Amending commits
- Stashing
- Discarding changes
- Cloning & pushing to SSH remotes
  - (Right now only for HTTP remotes)
- Adding tags

… and some edge cases in existing code we wrote :(

Lessons Learned

- Document your code
- Issue tracking
- Writing async code
- Working with large legacy code base
- Writing tests
- Reading API documentation
- Internal workings of git
- Cool git commands
- Client-server routes, HTTP interaction

A thank you to:

- Facebook (California was nice, yo);
- Our mentor, Paul;
- Jae and Cannon!

Questions?