lab

lab title

Creating a NodeJS Development Environment with AWS Cloud9 V1.00

Course title

BackSpace Academy
AWS Certified Associate

Table of Contents

Contents

About the Lab

Please note that not all AWS services are supported in all regions. Please use the US-East-1 (North Virginia) region for this lab.

These lab notes are to support the AWS Cloud9 lab in the Setting Up section of the AWS Certified Developer Associate Course.

Please note that AWS services change on a weekly basis and it is extremely important you check the version number on this document to ensure you have the lastest version with any updates or corrections.

Granting IAM Access to AWS Cloud9

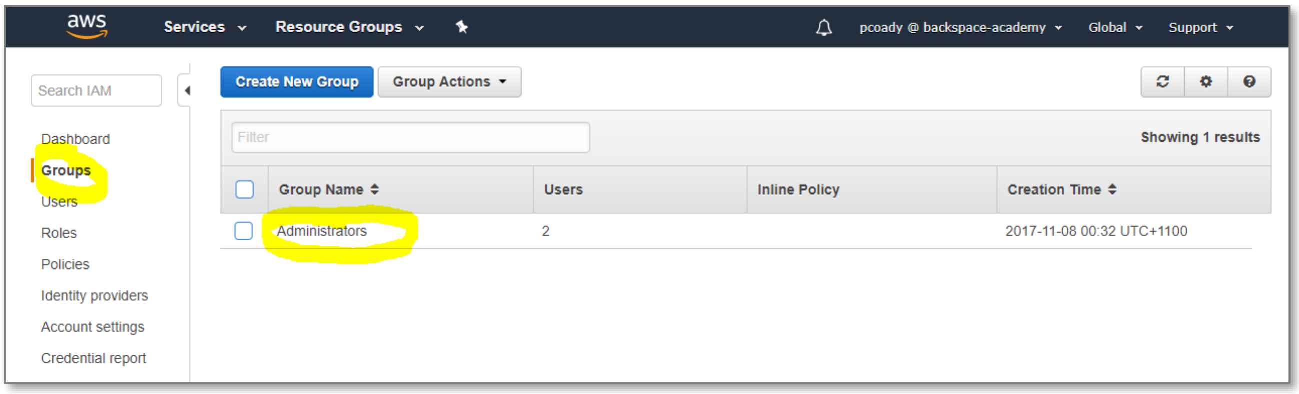
In this section, we will use the Identity and Access Management (IAM) service to add permissions for a group to access AWS Cloud9.

From the AWS console click “Services”

Select “IAM” from the Security, Identity & Compliance services.

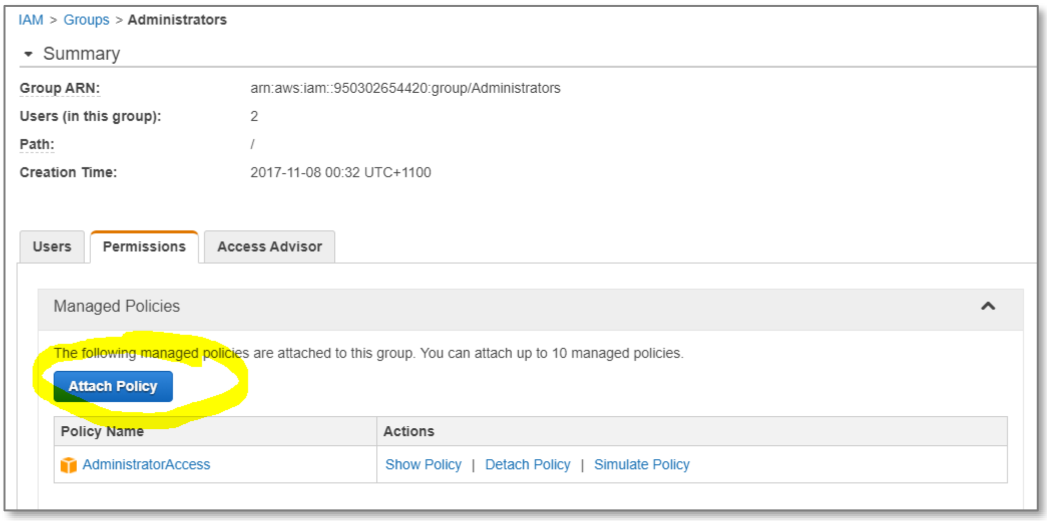
Select “Groups”

Select the Group you would like to add Cloud9 Permissions



	Group Name	Users	Inline Policy	Creation Time
☐	Administrators	2		2017-11-08 00:32 UTC+1100

Click “Attach Policy”



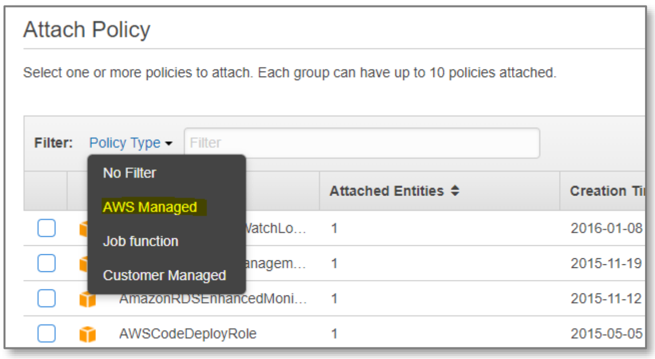
Group ARN: arn:aws:iam::950302654420:group/Administrators
Users (in this group): 2
Path: /
Creation Time: 2017-11-08 00:32 UTC+1100

The following managed policies are attached to this group. You can attach up to 10 managed policies.

Policy Name	Actions
AdministratorAccess	Show Policy | Detach Policy | Simulate Policy

Select “Policy Type”

Click “AWS Managed”



Attach Policy

Select one or more policies to attach. Each group can have up to 10 policies attached.

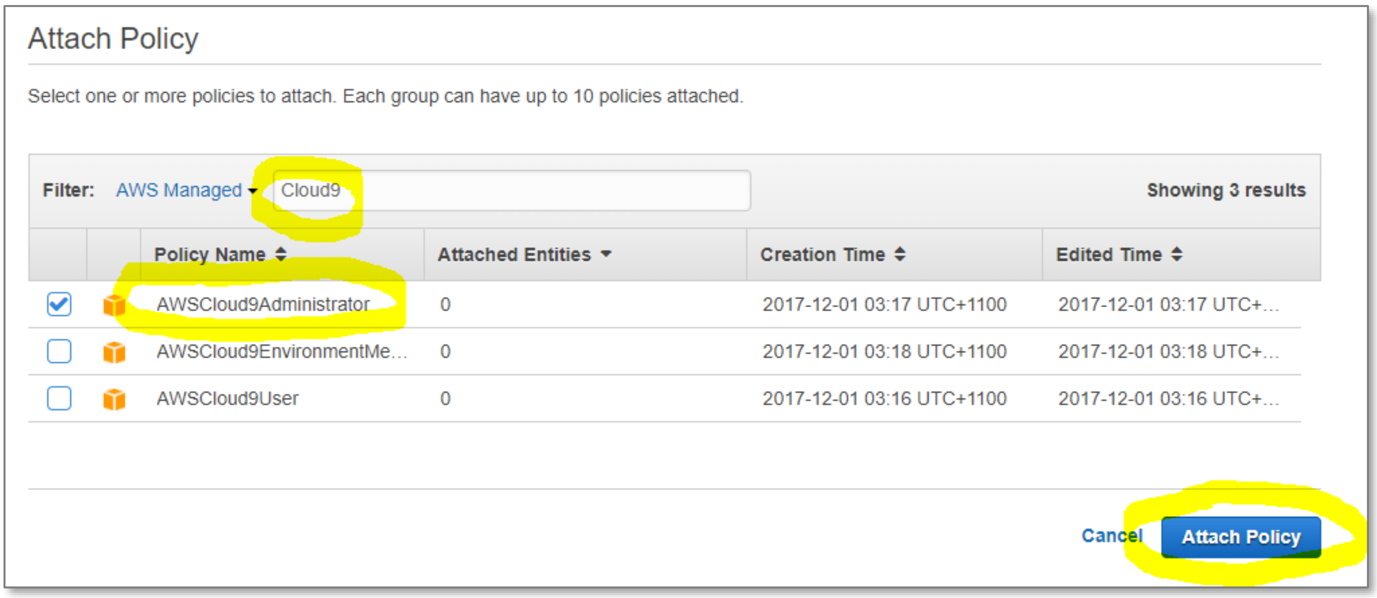
Filter: Policy Type ▾ Filter

			Attached Entities	Creation Ti
☐		…WatchLo…	1	2016-01-08
☐		…anagem…	1	2015-11-19
☐		AmazonRDSEnhancedMoni…	1	2015-11-12
☐		AWSCodeDeployRole	1	2015-05-05

Filter for “Cloud9”

Select “AWSCloud9Administrator”

Click “Attach Policy”



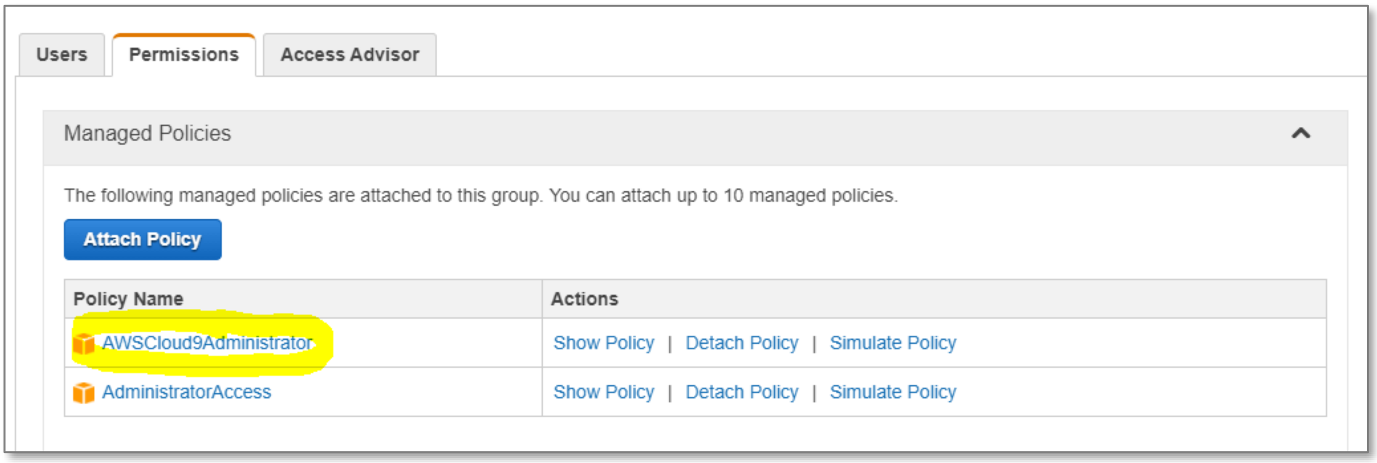
Attach Policy

Select one or more policies to attach. Each group can have up to 10 policies attached.

Filter: AWS Managed ▾ Cloud9 Showing 3 results

		Policy Name	Attached Entities	Creation Time	Edited Time
✓		AWSCloud9Administrator	0	2017-12-01 03:17 UTC+1100	2017-12-01 03:17 UTC+…
☐		AWSCloud9EnvironmentMe…	0	2017-12-01 03:18 UTC+1100	2017-12-01 03:18 UTC+…
☐		AWSCloud9User	0	2017-12-01 03:16 UTC+1100	2017-12-01 03:16 UTC+…

Your group will now have Administrator access to Cloud9


Users | Permissions | Access Advisor

Managed Policies

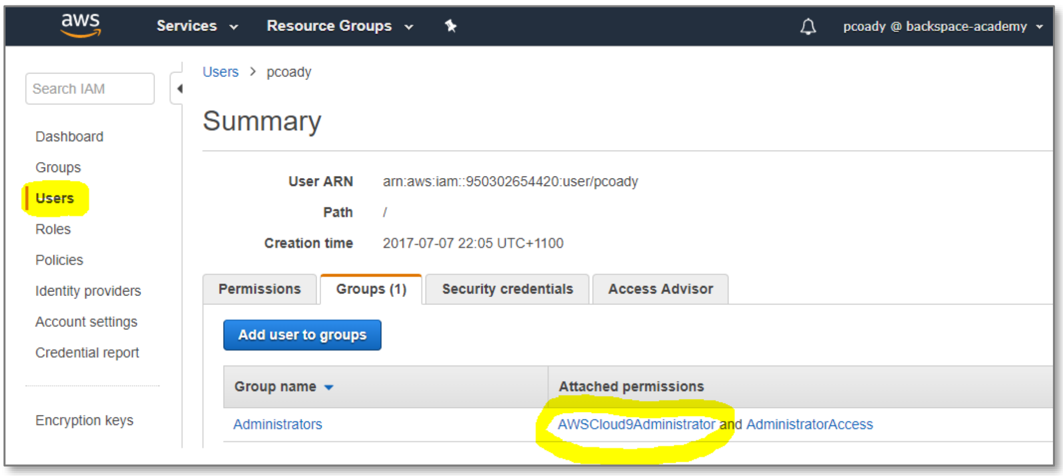
The following managed policies are attached to this group. You can attach up to 10 managed policies.

Attach Policy

Policy Name	Actions
AWSCloud9Administrator	Show Policy | Detach Policy | Simulate Policy
AdministratorAccess	Show Policy | Detach Policy | Simulate Policy

Go to “Users” and select your user that is part of the group.

You will see your user has inherited Cloud9 permission from the group.



Summary

User ARN arn:aws:iam::950302654420:user/pcoady
Path /
Creation time 2017-07-07 22:05 UTC+1100

Permissions | Groups (1) | Security credentials | Access Advisor

Add user to groups

Group name	Attached permissions
Administrators	AWSCloud9Administrator and AdministratorAccess

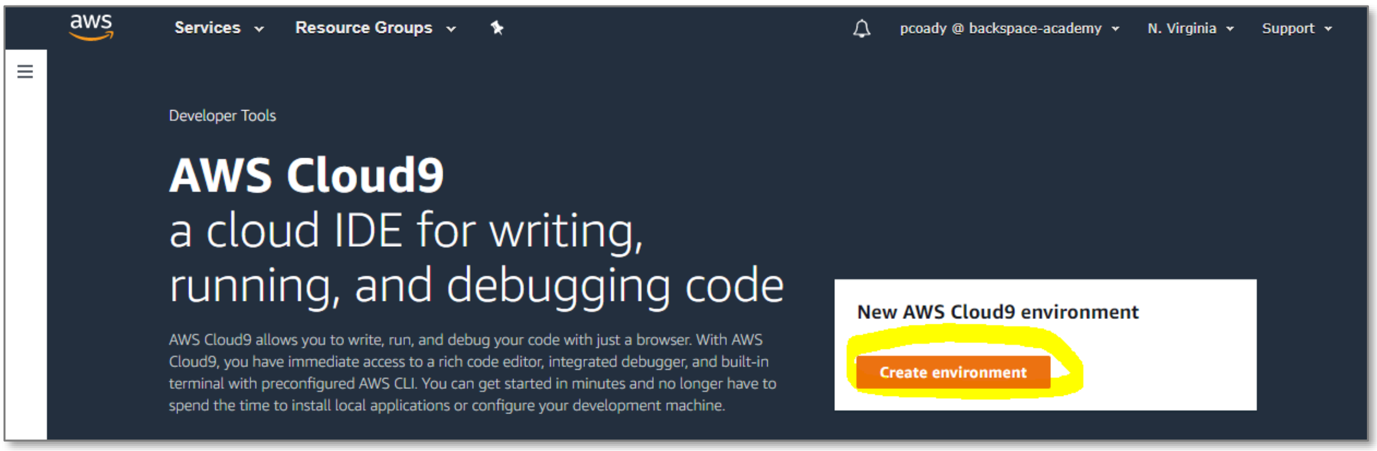
Creating a Cloud9 Development Environment on EC2

In this section, we will use the Cloud9 service to create a development environment on an EC2 instance.

From the AWS console click “Services”

Click ‘Cloud9”

Click “Create Environment”

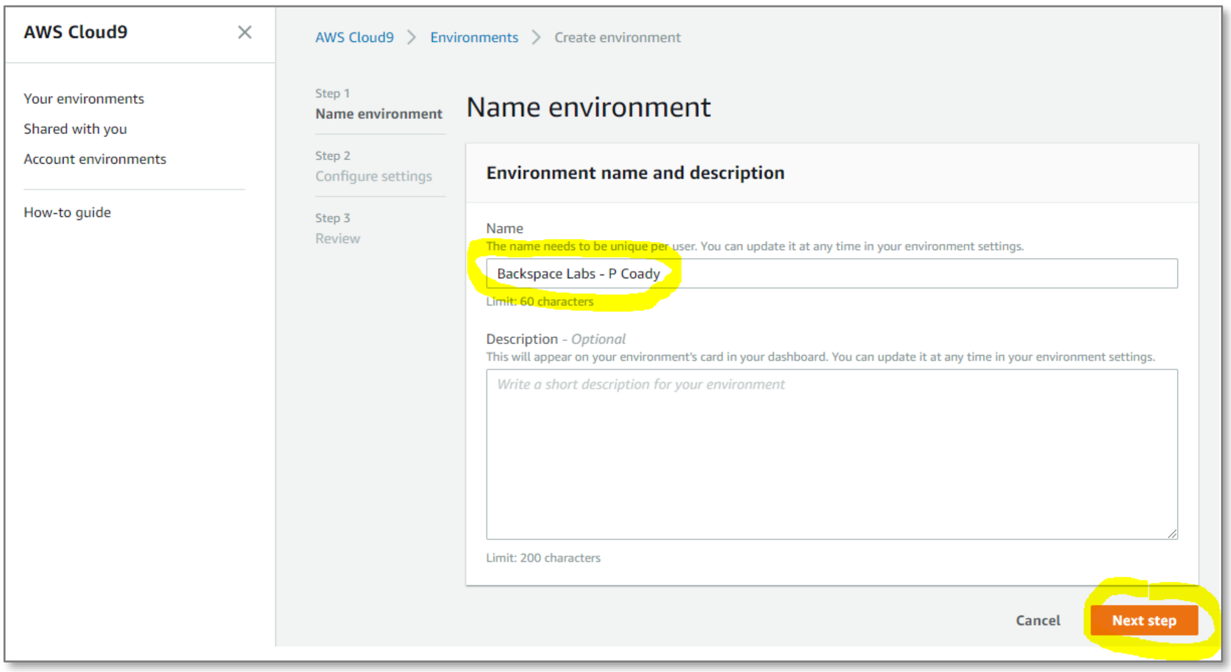


AWS Cloud9
a cloud IDE for writing, running, and debugging code

AWS Cloud9 allows you to write, run, and debug your code with just a browser. With AWS Cloud9, you have immediate access to a rich code editor, integrated debugger, and built-in terminal with preconfigured AWS CLI. You can get started in minutes and no longer have to spend the time to install local applications or configure your development machine.

Give your environment a unique name.

Click “Next Step”



AWS Cloud9

Your environments
Shared with you
Account environments
How-to guide

AWS Cloud9 > Environments > Create environment

Step 1 Name environment
Step 2 Configure settings
Step 3 Review

Name environment

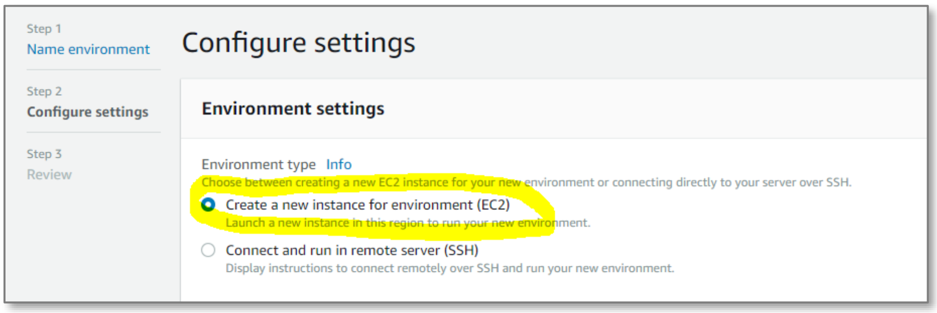
Environment name and description

Name
The name needs to be unique per user. You can update it at any time in your environment settings.
Backspace Labs - P Coady
Limit: 60 characters

Description - *Optional*
This will appear on your environment's card in your dashboard. You can update it at any time in your environment settings.
Write a short description for your environment
Limit: 200 characters

Cancel | Next step

Select EC2 environment.



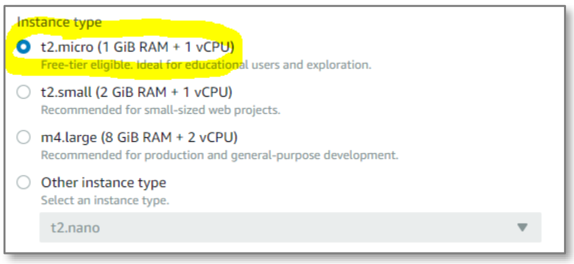
Step 1 Name environment
Step 2 Configure settings
Step 3 Review

Configure settings

Environment settings

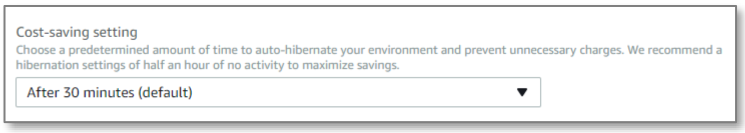
Environment type Info
Choose between creating a new EC2 instance for your new environment or connecting directly to your server over SSH.
● Create a new instance for environment (EC2)
Launch a new instance in this region to run your new environment.
○ Connect and run in remote server (SSH)
Display instructions to connect remotely over SSH and run your new environment.

Select t2 micro to stay in the free tier



Instance type
● t2.micro (1 GiB RAM + 1 vCPU)
Free-tier eligible. Ideal for educational users and exploration.
○ t2.small (2 GiB RAM + 1 vCPU)
Recommended for small-sized web projects.
○ m4.large (8 GiB RAM + 2 vCPU)
Recommended for production and general-purpose development.
○ Other instance type
Select an instance type.
t2.nano

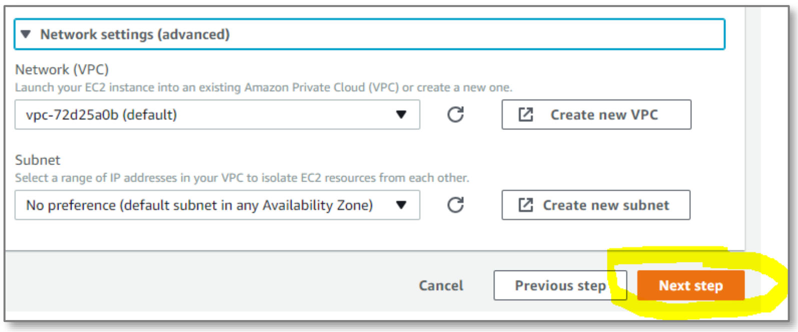
Leave hibernation setting at 30 mins



Cost-saving setting
Choose a predetermined amount of time to auto-hibernate your environment and prevent unnecessary charges. We recommend a hibernation settings of half an hour of no activity to maximize savings.
After 30 minutes (default)

Leave Network settings as default

Click “Next Step”

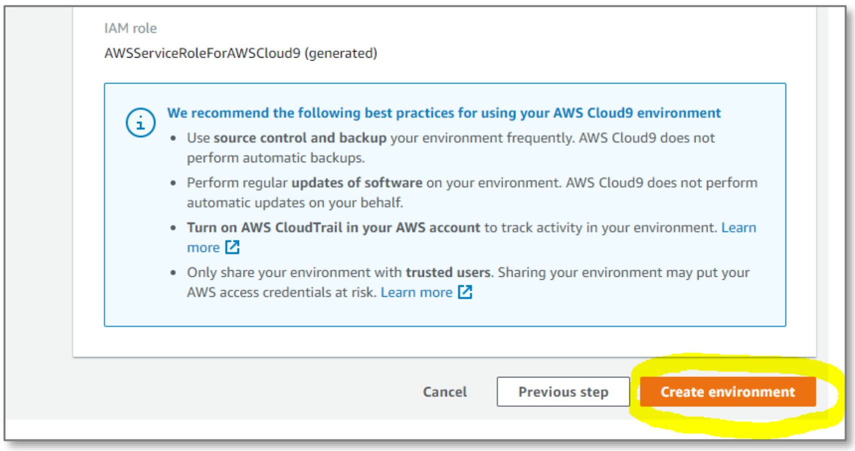


▼ Network settings (advanced)

Network (VPC)
Launch your EC2 instance into an existing Amazon Private Cloud (VPC) or create a new one.
vpc-72d25a0b (default)

Subnet
Select a range of IP addresses in your VPC to isolate EC2 resources from each other.
No preference (default subnet in any Availability Zone)

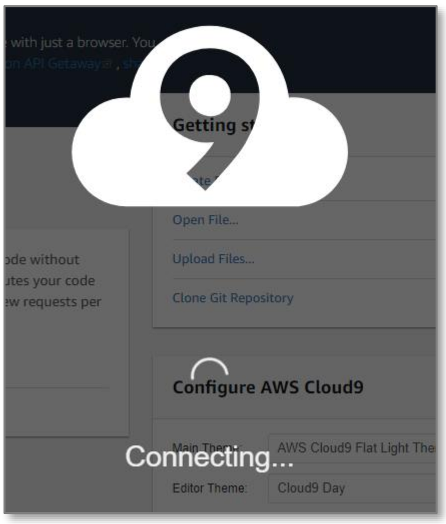
Click “Create Environment”



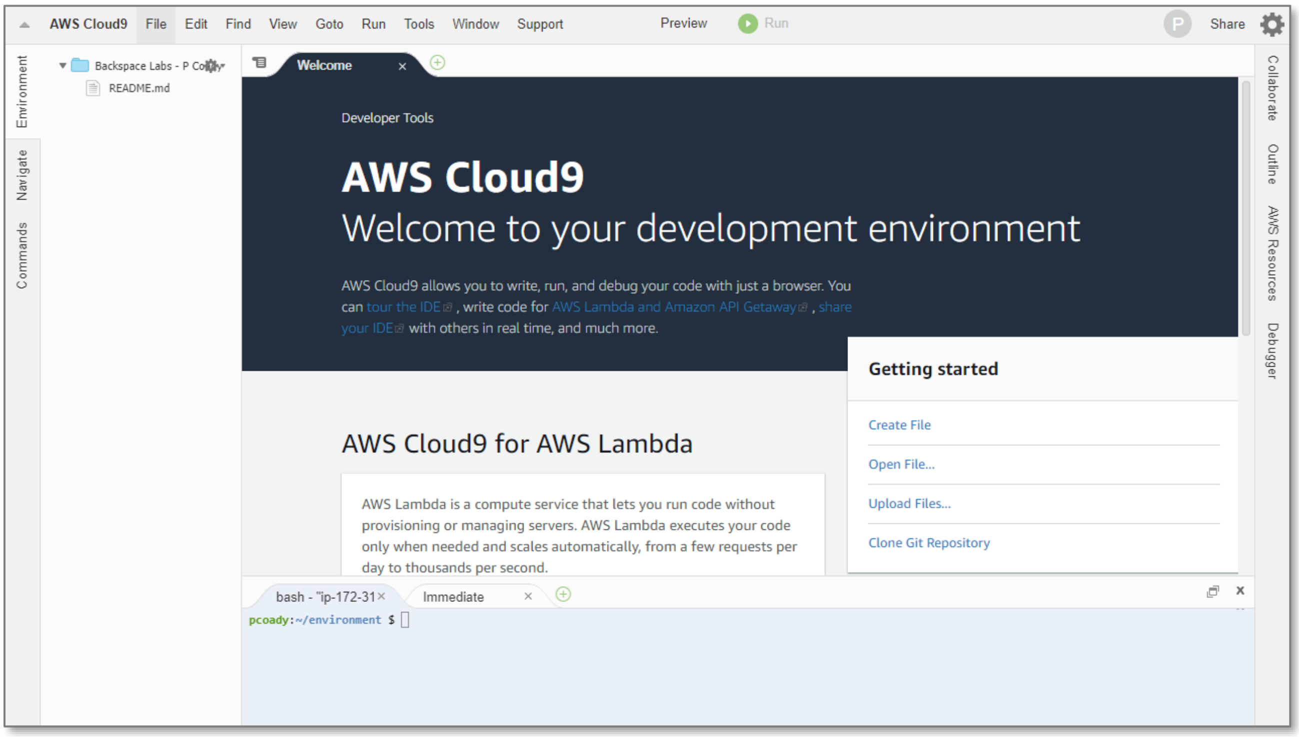
IAM role
AWSServiceRoleForAWSCloud9 (generated)

We recommend the following best practices for using your AWS Cloud9 environment
- Use **source control and backup** your environment frequently. AWS Cloud9 does not perform automatic backups.
- Perform regular **updates of software** on your environment. AWS Cloud9 does not perform automatic updates on your behalf.
- **Turn on AWS CloudTrail in your AWS account** to track activity in your environment. Learn more
- Only share your environment with **trusted users**. Sharing your environment may put your AWS access credentials at risk. Learn more

The environment creation process will begin



After some time, your environment will be created.



Developer Tools

AWS Cloud9

Welcome to your development environment

AWS Cloud9 allows you to write, run, and debug your code with just a browser. You can tour the IDE, write code for AWS Lambda and Amazon API Getaway, share your IDE with others in real time, and much more.

Getting started

Create File

Open File...

Upload Files...

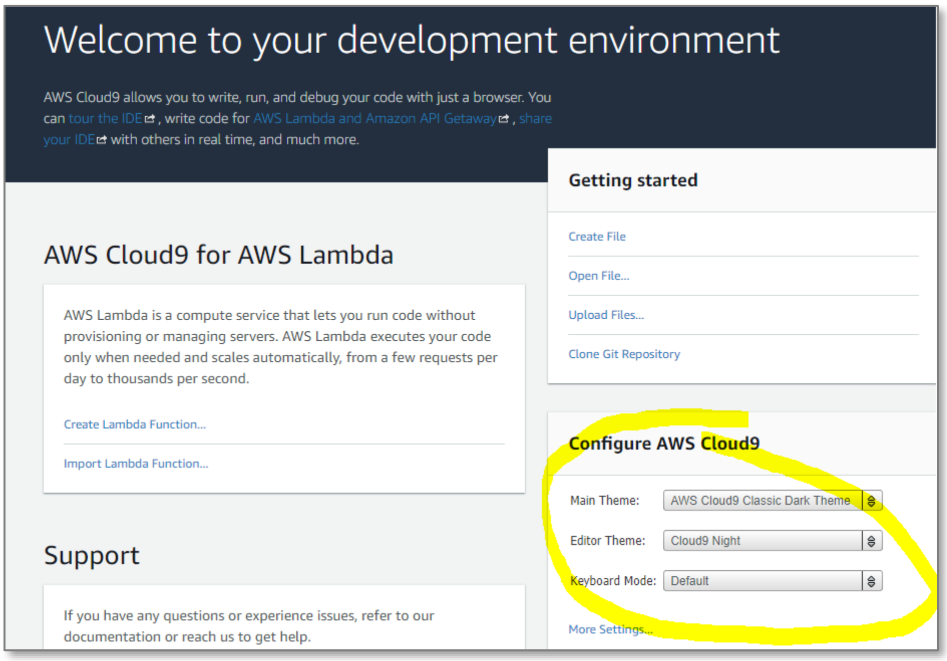
Clone Git Repository

AWS Cloud9 for AWS Lambda

AWS Lambda is a compute service that lets you run code without provisioning or managing servers. AWS Lambda executes your code only when needed and scales automatically, from a few requests per day to thousands per second.

```
pcoady:~/environment $
```

You can customize the look and feel of the IDE.



Welcome to your development environment

AWS Cloud9 allows you to write, run, and debug your code with just a browser. You can tour the IDE, write code for AWS Lambda and Amazon API Getaway, share your IDE with others in real time, and much more.

Getting started

Create File

Open File...

Upload Files...

Clone Git Repository

AWS Cloud9 for AWS Lambda

AWS Lambda is a compute service that lets you run code without provisioning or managing servers. AWS Lambda executes your code only when needed and scales automatically, from a few requests per day to thousands per second.

Create Lambda Function...

Import Lambda Function...

Configure AWS Cloud9

Main Theme: AWS Cloud9 Classic Dark Theme

Editor Theme: Cloud9 Night

Keyboard Mode: Default

More Settings...

Support

If you have any questions or experience issues, refer to our documentation or reach us to get help.

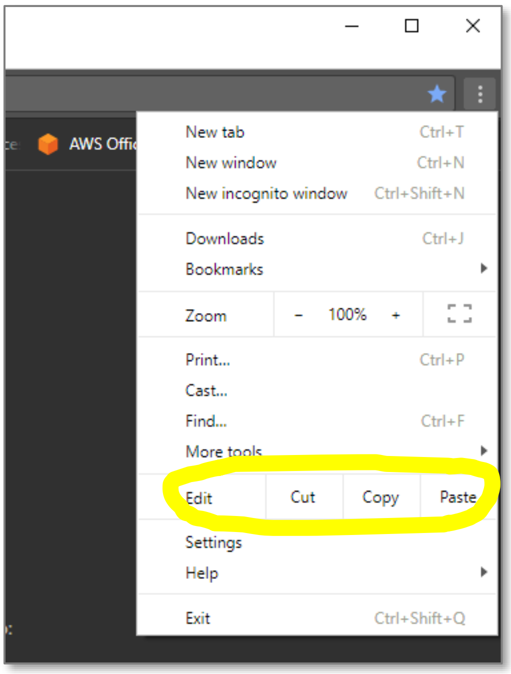
Sending Commands to a Cloud9 EC2 Instance

In this section, we will use the Cloud9 service to send Linux commands to the Cloud9 EC2 instance.

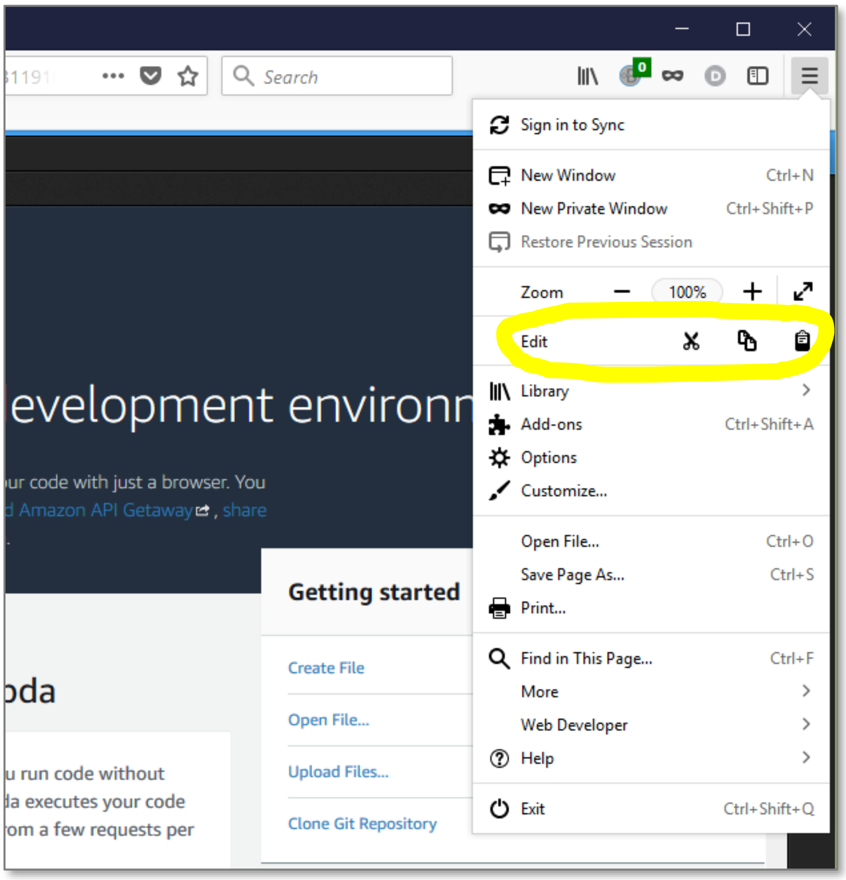
Please note:

Cut and Paste (right click or Cloud9 menu) may not work directly in Cloud9. If you cannot paste into Cloud9 then use the browser paste menu item or use ctrl-v (Windows) / cmd-v (MAC).

e.g. for Chrome:



e.g. for Firefox

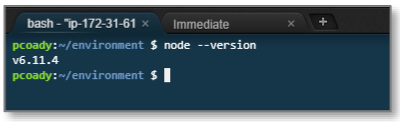


Sending Commands to the EC2 Instance

At the bottom of the screen will be the Linux terminal console pane.

Check that NodeJS is already installed.

```
node --version
```



```
pcoady:~/environment $ node --version
v6.11.4
pcoady:~/environment $
```

Update Amazon Linux (CentOS) operating system

```
sudo yum update -y
```

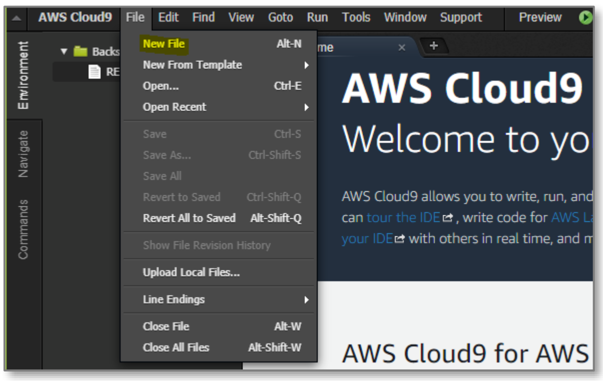
Install the AWS Javascript SDK

```
npm install aws-sdk
```

Running Code on your Cloud9 EC2 Instance

In this section, we will use the Cloud9 service to run NodeJS code the Cloud9 EC2 instance.

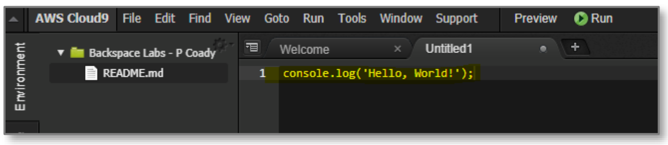
Select “File” – “New File”



AWS Cloud9 allows you to write, run, and
can tour the IDE, write code for AWS L
your IDE with others in real time, and m

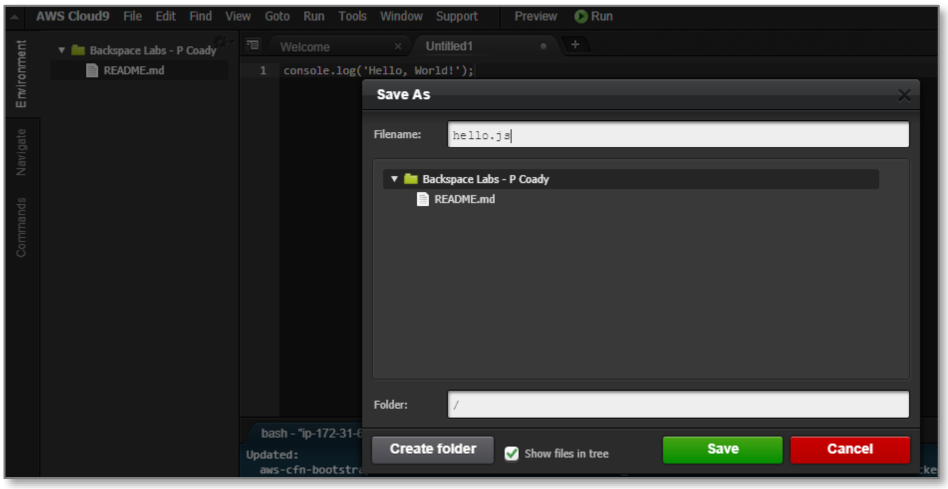
Create the classic “Hello World “ application

```
console.log('Hello World!');
```



```
console.log('Hello, World!');
```

Save as “hello.js”

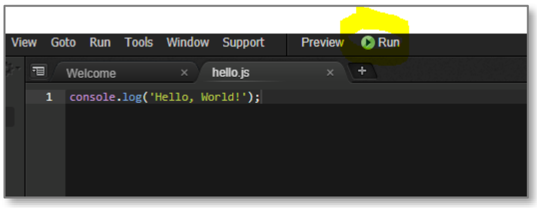


```
console.log('Hello, World!');
```

Filename: hello.js

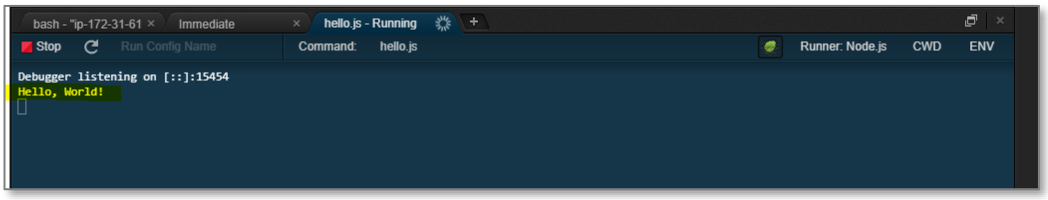
Folder: /

Click “Run”



```
console.log('Hello, World!');
```

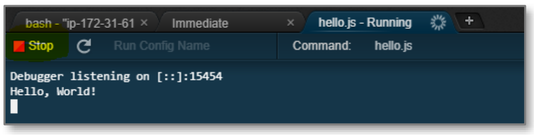
You can see the console output from your application



Command: hello.js

```
Debugger listening on [::]:15454
Hello, World!
```

Click “Stop” to stop the application



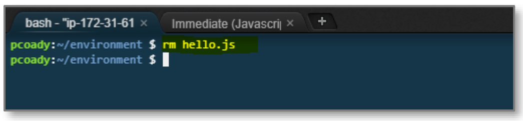
Command: hello.js

```
Debugger listening on [::]:15454
Hello, World!
```

Close the application tab to remove it from the IDE.

Go back to the Linux console and remove the hello.js file

```
rm hello.js
```



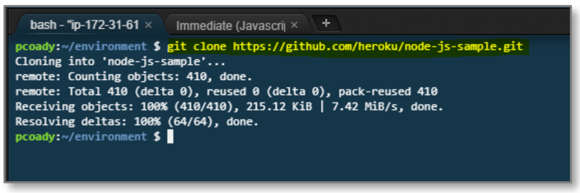
```
pcoady:~/environment $ rm hello.js
pcoady:~/environment $
```

Cloning a GitHub Repository to your Cloud9 EC2 Instance

In this section, we will use the Cloud9 service to clone a GitHub repository to the Cloud9 EC2 instance. We will then run the code in the repository.

Go back to the Linux console and clone the following sample NodeJS app repository.

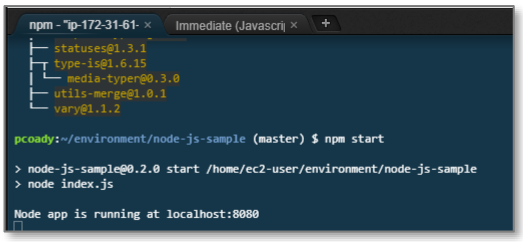
```
git clone https://github.com/heroku/node-js-sample.git
```



```
pcoady:~/environment $ git clone https://github.com/heroku/node-js-sample.git
Cloning into 'node-js-sample'...
remote: Counting objects: 410, done.
remote: Total 410 (delta 0), reused 0 (delta 0), pack-reused 410
Receiving objects: 100% (410/410), 215.12 KiB | 7.42 MiB/s, done.
Resolving deltas: 100% (64/64), done.
pcoady:~/environment $
```

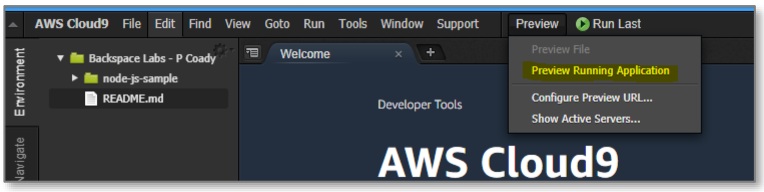
Now run the application.

```
cd node-js-sample
npm install
npm start
```

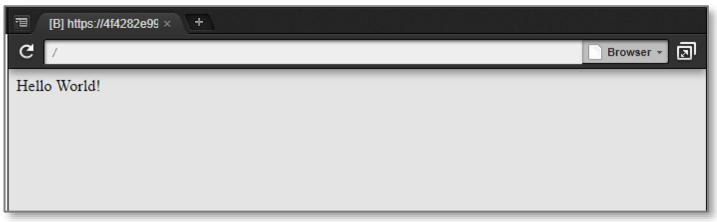


```
├── statuses@1.3.1
├┬ type-is@1.6.15
│ └── media-typer@0.3.0
├── utils-merge@1.0.1
└── vary@1.1.2

pcoady:~/environment/node-js-sample (master) $ npm start

> node-js-sample@0.2.0 start /home/ec2-user/environment/node-js-sample
> node index.js

Node app is running at localhost:8080
```

To see your running web application, click “Preview” – “Preview Running Application”



A new tab will open displaying the NodeJS web application in a web browser.



```
Hello World!
```

Editing Files

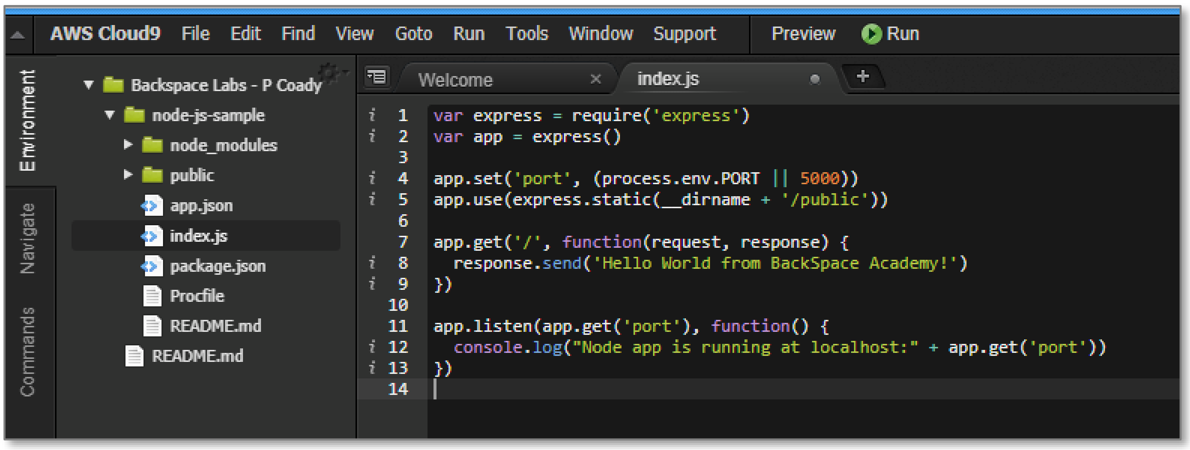
Press Ctr-C (Cmd-C for Mac) to stop the application.

Select the “Environment” tab to see the EC2 environment file system.

Expand the node-js-sample directory.

Double click “index.js” to open it for editing.

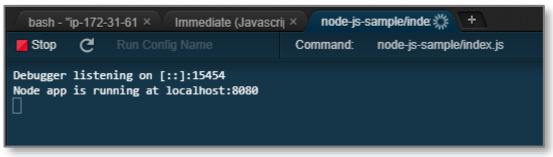
Change line 8 to give a different response message.



```
 1  var express = require('express')
 2  var app = express()
 3
 4  app.set('port', (process.env.PORT || 5000))
 5  app.use(express.static(__dirname + '/public'))
 6
 7  app.get('/', function(request, response) {
 8    response.send('Hello World from BackSpace Academy!')
 9  })
10
11  app.listen(app.get('port'), function() {
12    console.log("Node app is running at localhost:" + app.get('port'))
13  })
14
```

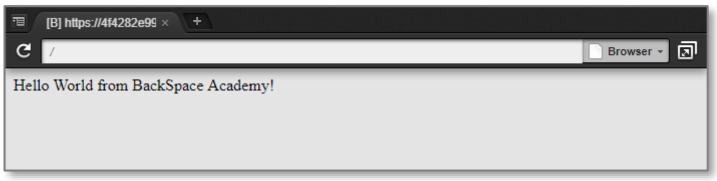
Click “Run” to run index.js in NodeJS.

The Linux terminal will show the app running again



```
Debugger listening on [::]:15454
Node app is running at localhost:8080
```

To see your new web application, click “Preview” – “Preview Running Application”



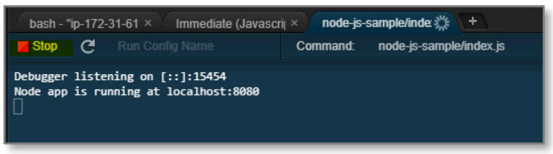
Hello World from BackSpace Academy!

Cleaning Up

Your Cloud 9 EC2 development environment will automatically go into hibernation after 30 mins of inactivity. You will not need to terminate it to save costs.

You should clean up the code in environment ready for the next lab.

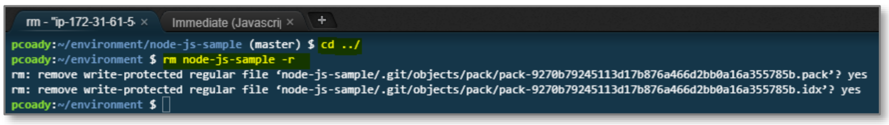
Stop the application.



```
Debugger listening on [::]:15454
Node app is running at localhost:8080
```

Go to the Linux terminal and remove the application

```
cd ../
rm node-js-sample -r
```



```
pcoady:~/environment/node-js-sample (master) $ cd ../
pcoady:~/environment $ rm node-js-sample -r
rm: remove write-protected regular file ‘node-js-sample/.git/objects/pack/pack-9270b79245113d17b876a466d2bb0a16a355785b.pack’? yes
rm: remove write-protected regular file ‘node-js-sample/.git/objects/pack/pack-9270b79245113d17b876a466d2bb0a16a355785b.idx’? yes
pcoady:~/environment $
```