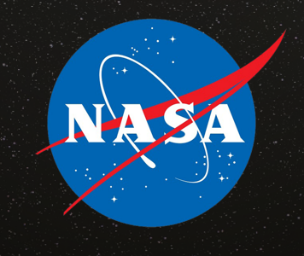


NASA EVA Path Finder User Manual

****

User Manual

**Maintained by the NASA EVA Path Finder project**

**Initially prepared by Phase 2**

**Nicoleta Florea, Bradley Hankinson, Brittany Ofori, George Wheeler, Justin Widmann**

User Manual

For

NASA EVA Path Finder

**Version 1.7**

This document will be maintained by the NASA EVA Path project.

Initially prepared by Phase 2

Nicoleta Florea, Bradley Hankinson, Brittany Ofori, George Wheeler, Justin Widmann

University of Maryland University College

SWEN 670 9040

Software Engineering Project

July 29, 2018

**Table of Contents**

[1. Overview 6](#_Toc520686886)

[1.1 Purpose 6](#_Toc520686887)

[1.2 Intended Audience 6](#_Toc520686888)

[1.3 Scope 6](#_Toc520686889)

[1.4 References 6](#_Toc520686890)

[2. Technical Stack 7](#_Toc520686891)

[2.1 Front End 7](#_Toc520686892)

[2.2 Back End 7](#_Toc520686893)

[2.3 Supplemental Tools and Add-ins 7](#_Toc520686894)

[3. Software Installation 8](#_Toc520686895)

[3.1 Windows Installation 8](#_Toc520686896)

[3.1.1 Step 1: Install Node.js 8](#_Toc520686897)

[3.1.2 Step 2: Install Yarn 8](#_Toc520686898)

[3.1.3 Step 3: Install or Update Java 8 9](#_Toc520686899)

[3.1.4 Step 4: Install Maven (Apache Maven Project) 10](#_Toc520686900)

[3.1.5 Step 5: Install Visual Studio 11](#_Toc520686901)

[3.1.6 Step 6: Install Python 11](#_Toc520686902)

[3.1.7 Step 7: Confirm File Locations 12](#_Toc520686903)

[3.1.8 Step 8: Clone GitHub Repository 12](#_Toc520686904)

[3.1.9 Step 9: Set Windows Environment Variables 12](#_Toc520686905)

[3.1.10 Step 10: Yarn Dependencies 13](#_Toc520686906)

[3.1.11 Step 11: Start Yarn (Front-End) 13](#_Toc520686907)

[3.1.12 Step 12: Start Yarn (Server) 14](#_Toc520686908)

[3.1.13 Step 13: Browser – Navigate to UI 15](#_Toc520686909)

[3.2 Ubuntu Installation 15](#_Toc520686910)

[3.2.1 Step 1: Install Node.js 15](#_Toc520686911)

[3.2.2 Step 2: Install Yarn 16](#_Toc520686912)

[3.2.3 Step 3: Install Java 16](#_Toc520686913)

[3.2.4 Step 4: Install Maven 16](#_Toc520686914)

[3.2.5 Step 5: GitHub 16](#_Toc520686915)

[3.2.6 Step 6: Yarn Dependencies 16](#_Toc520686916)

[3.2.7 Step 7: Yarn Start 16](#_Toc520686917)

[3.2.8 Step 8: Yarn Compile 17](#_Toc520686918)

[3.2.9 Step 9: Navigate to UI 17](#_Toc520686919)

[4. Dynamic Onboard Ubiquitous Graphics (DOUG) Software 18](#_Toc520686920)

[4.1 Overview 18](#_Toc520686921)

[4.2 Registration 18](#_Toc520686922)

[4.2.1 Step 1: Professor / Signatory Account 18](#_Toc520686923)

[4.2.2 Step 2: Submit Request for Software 18](#_Toc520686924)

[4.2.3 Step 3: Approval Confirmation 20](#_Toc520686925)

[4.2.4 Step 4: Download DOUG Installation 21](#_Toc520686926)

[4.2.5 Step 5: Activation Code 22](#_Toc520686927)

[4.3 Installation 23](#_Toc520686928)

[4.3.1 Step 1: Run DOUG Setup Installation 23](#_Toc520686929)

[4.3.2 Step 2: Paste Activation Code 23](#_Toc520686930)

[4.3.3 Step 3: Complete Wizard 24](#_Toc520686931)

[4.3.4 Step 4: Confirm Installation 24](#_Toc520686932)

[5. Getting Started 26](#_Toc520686933)

[5.1 Launching the Application 26](#_Toc520686934)

[5.1.1 Windows 26](#_Toc520686935)

[5.1.2 Ubuntu 26](#_Toc520686936)

[6. Navigating the Navigator 27](#_Toc520686937)

[6.1 Controls Tab 27](#_Toc520686938)

[6.2 Upload Files Tab 28](#_Toc520686939)

[6.3 Path Results Tab 29](#_Toc520686940)

[6.4 Overall Model 29](#_Toc520686941)

[7. Common Errors and Fixes 30](#_Toc520686942)

[7.1 Compile Exit Code 1 30](#_Toc520686943)

[7.2 Model Time Out 30](#_Toc520686944)

[7.2.1 Fix One: 30](#_Toc520686945)

[7.2.2 Fix Two: 31](#_Toc520686946)

[7.2.3 Fix Three (Pending): 31](#_Toc520686947)

[8. Packages and Libraries 32](#_Toc520686948)

[8.1 Create a Package – package.json 32](#_Toc520686949)

[8.2 Install NPM Package 34](#_Toc520686950)

[9. Appendix A: Glossary 35](#_Toc520686951)

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
| George Wheeler | 03/27/2018 | Combined installation documents into User Manual - Bradley Hankinson (Windows), - Nicoleta Florea (Ubuntu), and added the JSON and NPM instructions – Nicoleta Florea | 1.1 |
| George Wheeler | 03/31/2018 | Changed Outline of Document | 1.1 |
| Brad Hankinson | 04/01/2018 | Added screenshot of common error exit code 1, updated steps in Windows installation to omit server commands, and made minor formatting changes. | 1.2 |
| Brad Hankinson | 04/09/2018 | Added DOUG Software Registration and Installation sections | 1.3 |
| George Wheeler | 04/17/2018 | Made text formatting uniform throughout document. | 1.4 |
| Brad Hankinson | 04/20/2018 | Updated format, redacted contact information from DOUG screenshots, consolidated technical stack sections, and removed deprecated installation steps. | 1.5 |
| Nikki Florea | 4/21/2018 | Added Getting Started, Using the System, and Packages and Libraries sections, and added model loading sections to the Common Errors section | 1.6 |

# Overview

## Purpose

The purpose of this document is to serve as a singular reference to all things NASA EVA Path Finder for both first-time users and developers alike. Filling the roles of both a quick reference and general guide for managing the software, this document should evolve in parallel with the project throughout each phase.

## Intended Audience

This document was created for users and developers. It contains general information ranging from installation and configuration to troubleshooting.

Note: Enable Navigation Pane for outline navigation of the document by section.

## Scope

The scope of this document will expand with the requirements of the Path Finder project. It is intended to sufficiently describe installation, configuration, execution, and maintenance of the software for new users. Where appropriate, descriptions of functionality or specific feature details may be included.

As the software grows over time, new features that need supplemental instruction should be refactored into this resource. As this project reaches completion, this document will become the singular reference for installation and use of the software.

## References

Google Drive. (2017). NASA\_Path\_SRS\_milestone3. Retrieved from <https://drive.google.com/drive/folders/1eLc_dxzsFKtd2MFow875MQICLlQenDCQ>

Google Drive. (2017). NASA\_Path\_Phase\_2\_Installation\_Steps\_Milestone2. Retrieved from <https://drive.google.com/drive/folders/1YBU0MWOCIDdFLKCbpWmRvhSZG1m_8L7x>

# Technical Stack

## Front End

Jest – Test Execution

NodeJS – Package Management

React – UI Rendering Library

ThreeJS – 3D Rendering

Webpack – Module Bundler

Yarn – Package Management

## Back End

AJAX – Asynchronous Algorithm Display

Java – Software Platform

## Supplemental Tools and Add-ins

CircleCI – Continuous Integration

Docker – Web Building

Electron – Build Management

GitHub – Code Repository

Log4j2 – Logging Framework

Python – Code Base

Visual Studio - .NET Framework

# Software Installation

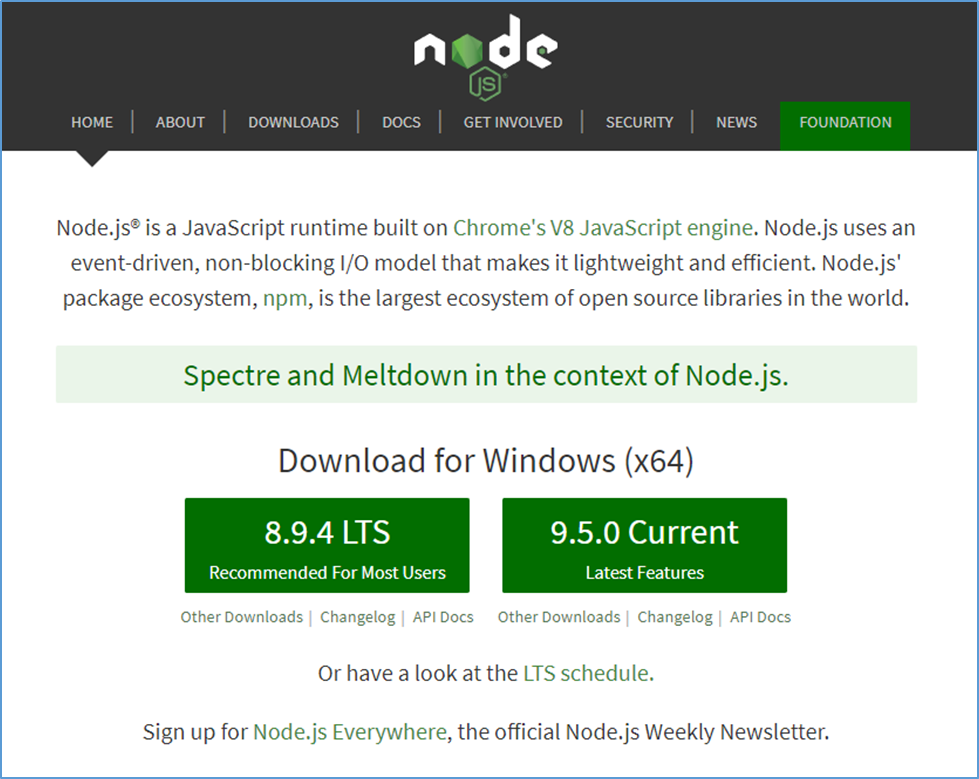
## Windows Installation

The following steps describe the process of installing the NASA Path Finder software for the Windows OS. Command descriptions and example screenshots were taken in Windows 10.

### Step 1: Install Node.js

Download location: <https://nodejs.org/en/>

Install to preferred location.



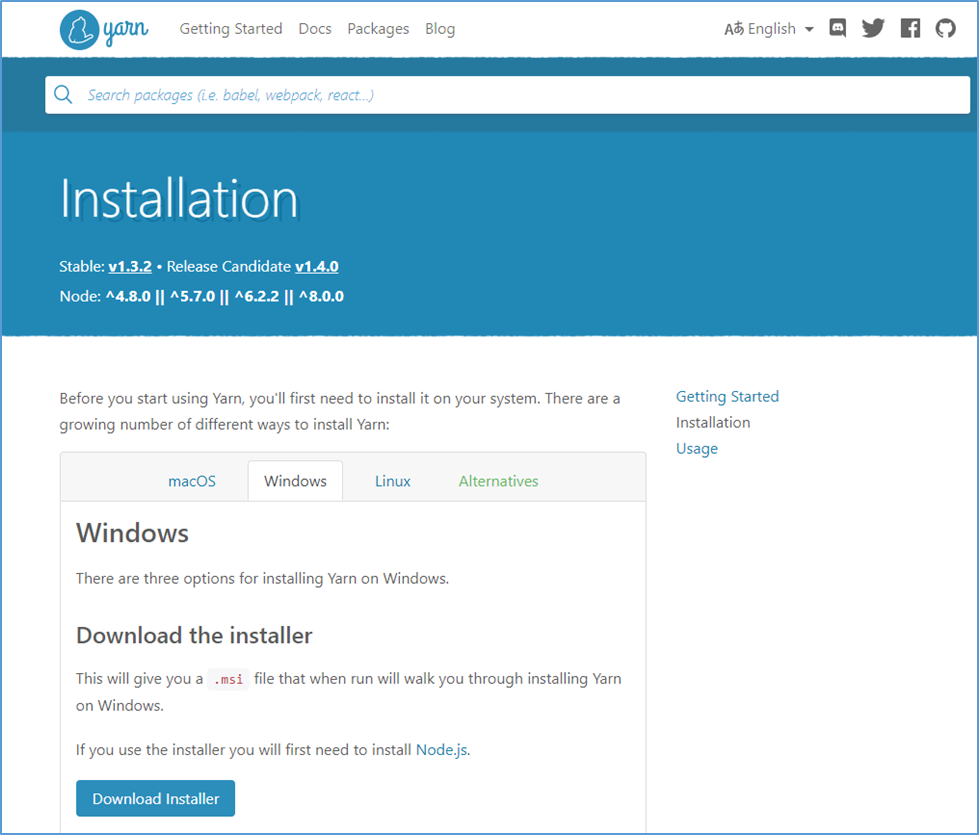
### Step 2: Install Yarn

Yarn is used to manage dependencies and execute the NASA Path software locally.

Download location: <https://yarnpkg.com/en/docs/install>

Select the Windows.msi installation download. As shown below, Node.js should be installed before completing this step.

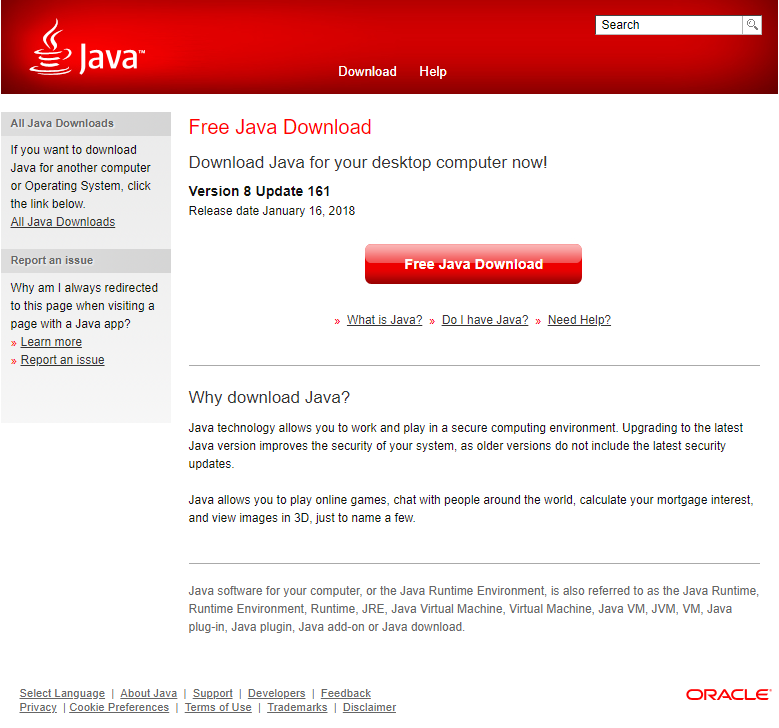
Use “Download Installer” and install at your preferred path.



### Step 3: Install or Update Java 8

Download location: <https://www.java.com/en/download/> (Version 8 Update 161 at the time of this document).

Install to preferred location.

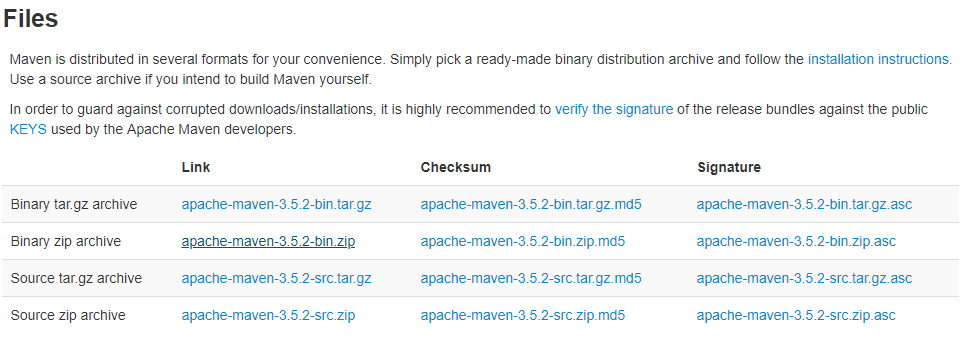


### Step 4: Install Maven (Apache Maven Project)

Download Location: <https://maven.apache.org/> (Version 3.5.2 at the time of this document).

Maven does not “install,” it’s a zipped folder that can be placed at your preferred location. Examples below place the folder in C:\.





### Step 5: Install Visual Studio

Install Visual Studio for .NET Framework support. A free, community version is available.

Download Location: <https://www.visualstudio.com/downloads/>

### Step 6: Install Python

Install Python.

Download Location: <https://www.python.org/downloads/>

### Step 7: Confirm File Locations

Assuming you are using C:\ as your default location, copy the following folders here.

Place unzipped “apache-maven-3.5.2” installation folder in C:\

### Step 8: Clone GitHub Repository

Clone latest GitHub repository for a local copy.

* Navigate to https://github.com/xpaddict/nasa-path-finder
* Click “Clone or Download” button for repository URL
* Open command line window and navigate to target parent directory
* Command: git clone <PasteURL>

### Step 9: Set Windows Environment Variables

Access Windows environment variables via Control Panel. The search can be used with “environment variables” to provide a link to this configuration.

Note: Be sure to select environment variables for the system and not individual users.

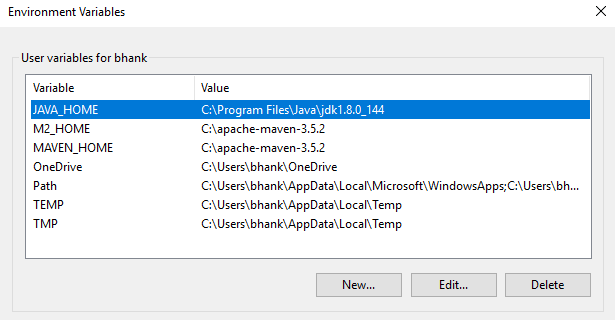
Again, the following paths assume a default location of C:\, if you chose another location be sure to set the paths relative to that location.

If the following 3 paths do not exist, create them by clicking “New”:

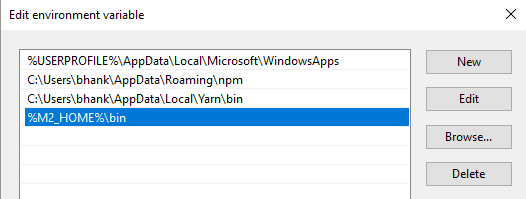
1. JAVA\_HOME – Locate the JDK folder where you installed Java 8

2. M2\_HOME – Locate the apache maven folder

3. MAVEN\_HOME – Locate the same apache maven folder



Select the “Path” Variable, Edit, and add “%M2\_HOME%\bin”

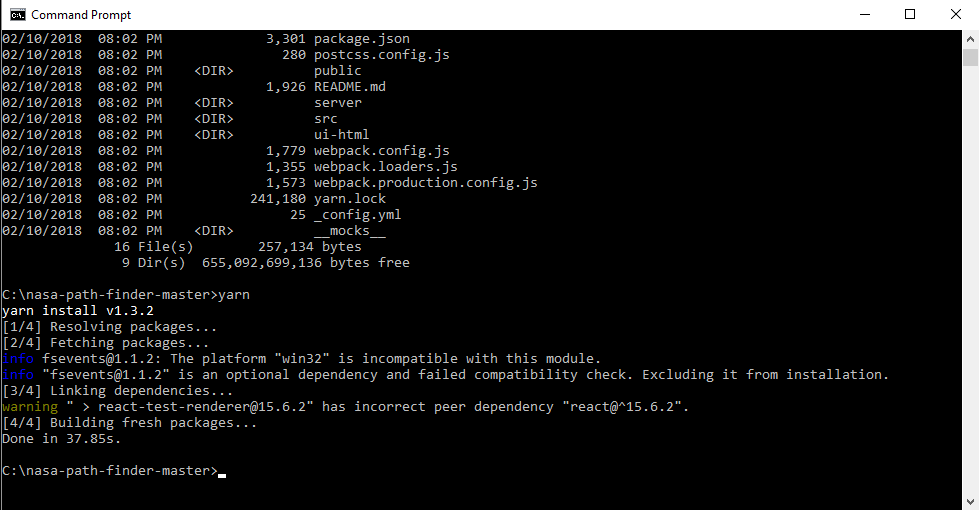


### Step 10: Yarn Dependencies

Update all yarn dependencies.

* Open a command window.
* Navigate to the /nasa-path-finder project directory and run “yarn” to download project dependencies. It may take a few minutes to complete the dependency downloads.

If you placed the folder in the C:\ directory, that would look like this:

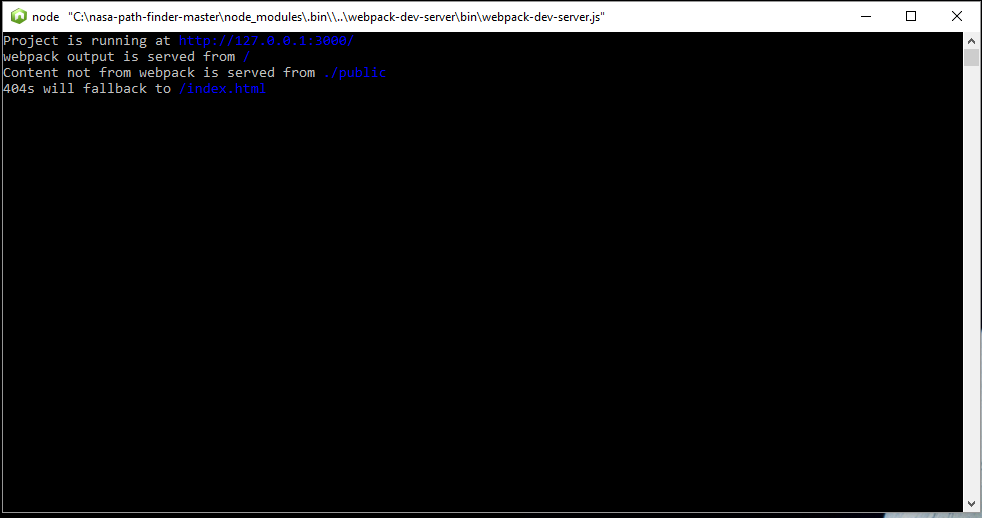


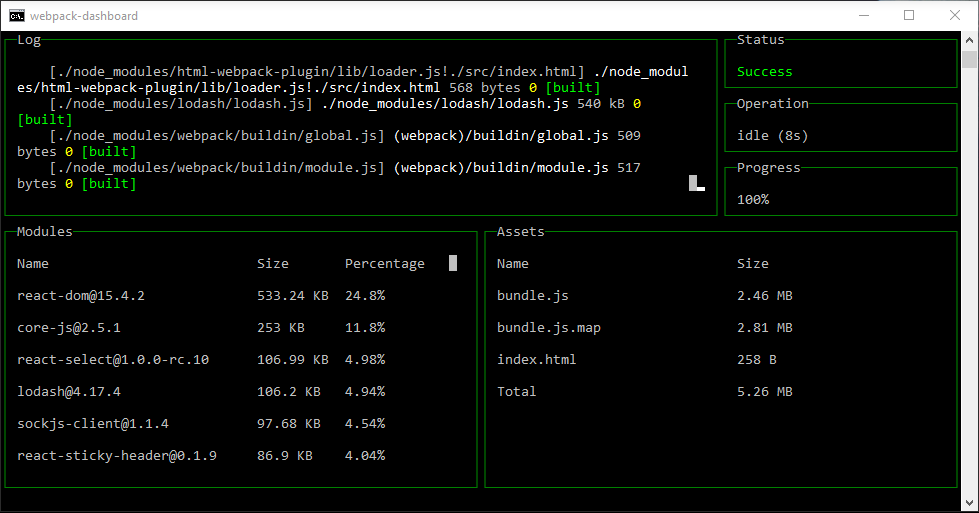
### Step 11: Start Yarn (Front-End)

Execute software front end.

* Open a command window.
* Navigate to the /nasa-path-finder project directory and run “yarn start” to execute the project at localhost:3000.

At this point two windows should open that look like this:



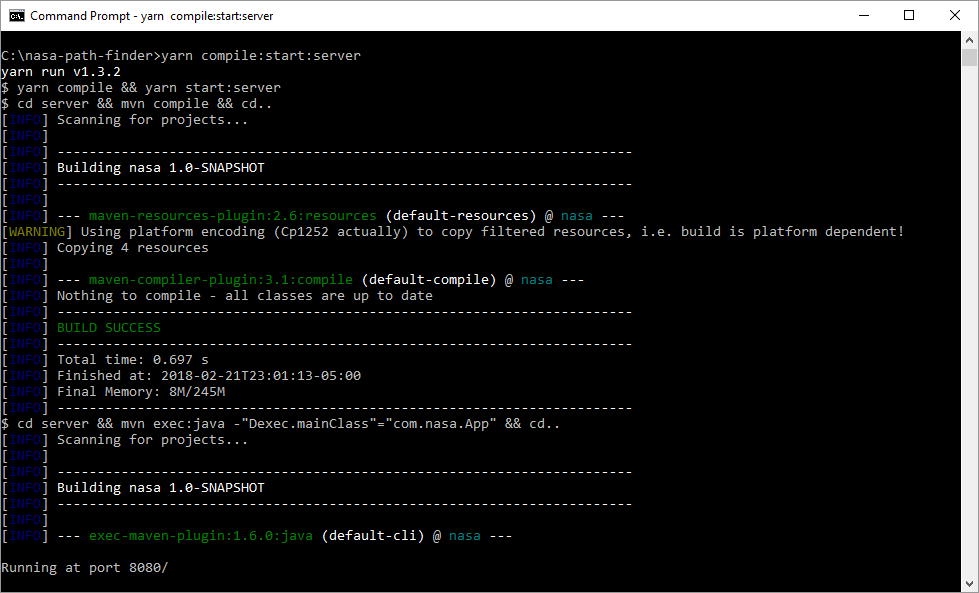


### Step 12: Start Yarn (Server)

Execute software back end.

* Leave the first two windows open and open a new command line window and navigate to the same project folder.
* Open another command window.
* Navigate to the /nasa-path-finder project directory and run “yarn compilewin:start:server”

It may take a few minutes and will download many files. Each execution after, it will look like this:



The last line should be “Running at port 8080/”

### Step 13: Browser – Navigate to UI

Open a browser and navigate to <http://127.0.0.1:3000> or <http://localhost:3000>

The tool should execute and load in the browser window:



## Ubuntu Installation

The following steps describe the process of installing the NASA Path Finder software for the Ubuntu OS.

### Step 1: Install Node.js

* sudo apt-get install nodejs

### Step 2: Install Yarn

* curl -sS https://dl.yarnpkg.com/debian/pubkey.gpg | sudo apt-key add -
* echo "deb https://dl.yarnpkg.com/debian/ stable main" | sudo tee

/etc/apt/sources.list.d/yarn.list

* sudo apt-get update && sudo apt-get install yarn

### Step 3: Install Java

* sudo apt-get install oracle-java8-installer

Check java version:

* java -version

### Step 4: Install Maven

* apt-cache search maven

Check maven version:

* mvn -version

Or to check that installed version is latest and greatest:

* sudo apt-get --only-upgrade install maven

### Step 5: GitHub

* Configure GitHub
* Create GitHub Account
* Install GitHub
* sudo apt-get update
* sudo apt-get install git
* Navigate to Repository

<https://github.com/xpaddict/nasa-path-finder>

Click green "Clone / Download" button and copy the URL it provides

* git clone <URL you just copied>

### Step 6: Yarn Dependencies

New terminal (*Ctrl+Alt+T*)

* cd nasa-path-finder
* yarn

If node error happens, use:

* yarn –ignore-engines

### Step 7: Yarn Start

New terminal (*Ctrl+Alt+T*)

* cd nasa-path-finder
* yarn start

### Step 8: Yarn Compile

New terminal (*Ctrl+Alt+T*)

* cd nasa-path-finder
* yarn compile:start:server

### Step 9: Navigate to UI

Open a browser and navigate to <http://127.0.0.1:3000> or <http://localhost:3000>

The tool should execute and load in the browser:



# Dynamic Onboard Ubiquitous Graphics (DOUG) Software

## Overview

This section describes the processes for application and installation of the DOUG modeling software. Access is provided on a machine-by-machine basis. So, all team members should apply for permission as early as possible in the semester phase.

## Registration

### Step 1: Professor / Signatory Account

Professor should sign in and confirm the signatory account is still active and can accept a new request. His account will serve as signatory authority for the registration request.

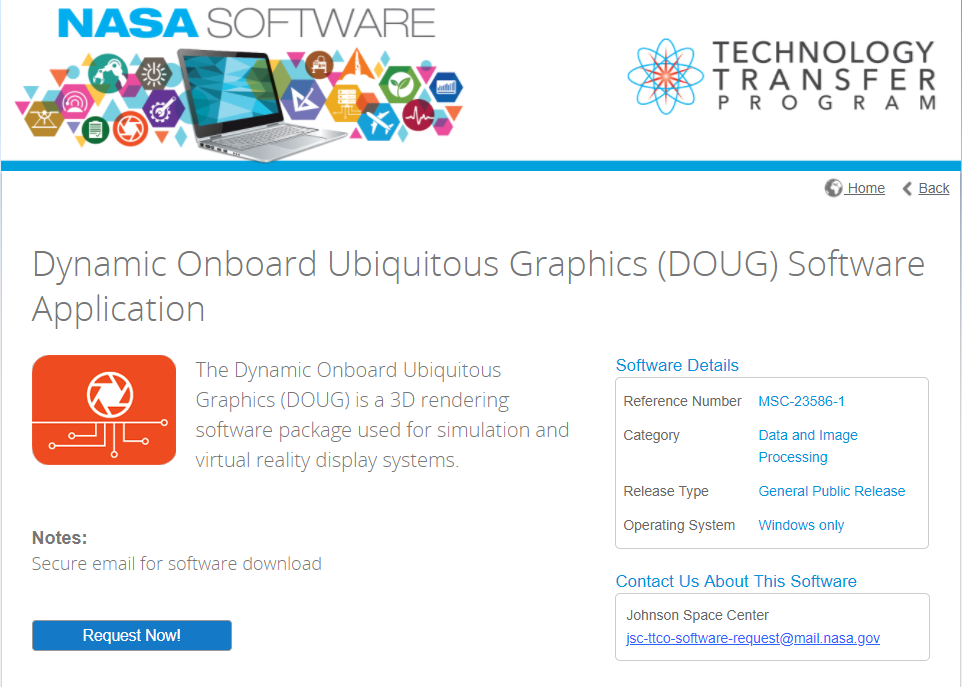
Current signatory authority is registered to:

Michael Brown

[michael.brown@umuc.edu](mailto:michael.brown@umuc.edu)

### Step 2: Submit Request for Software

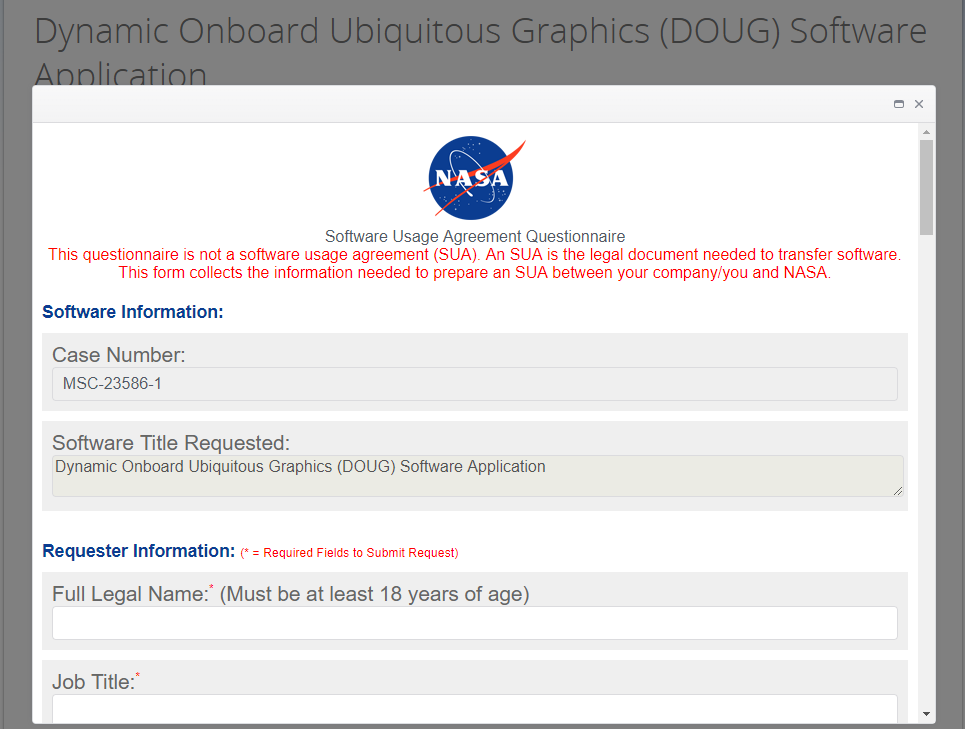
Navigate to http://software.nasa.gov and search applications for “DOUG”



Select “Request Now!”

Create a New Account.

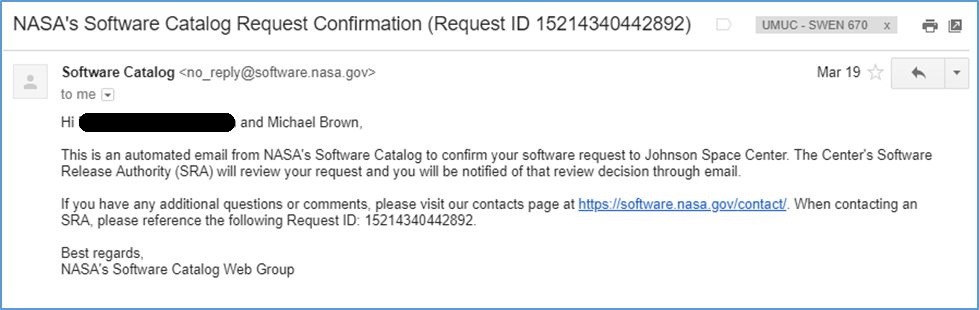
Complete Software Usage Agreement Questionnaire.



Enter professor’s information (Full Name and Business Email) as Signature Authority Contact.

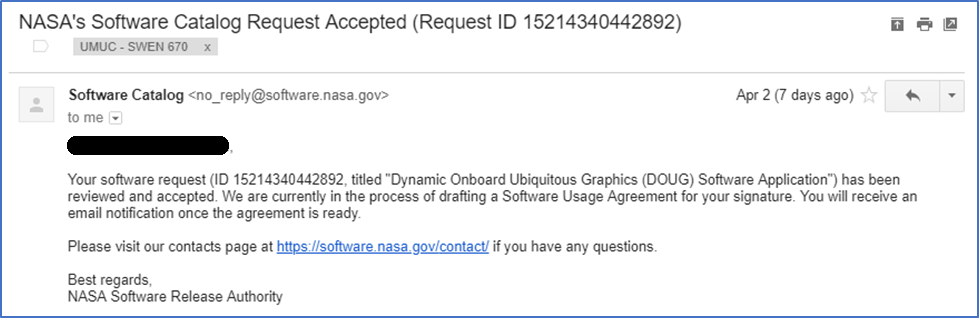


The following automated email confirmation will be sent to the registering email.

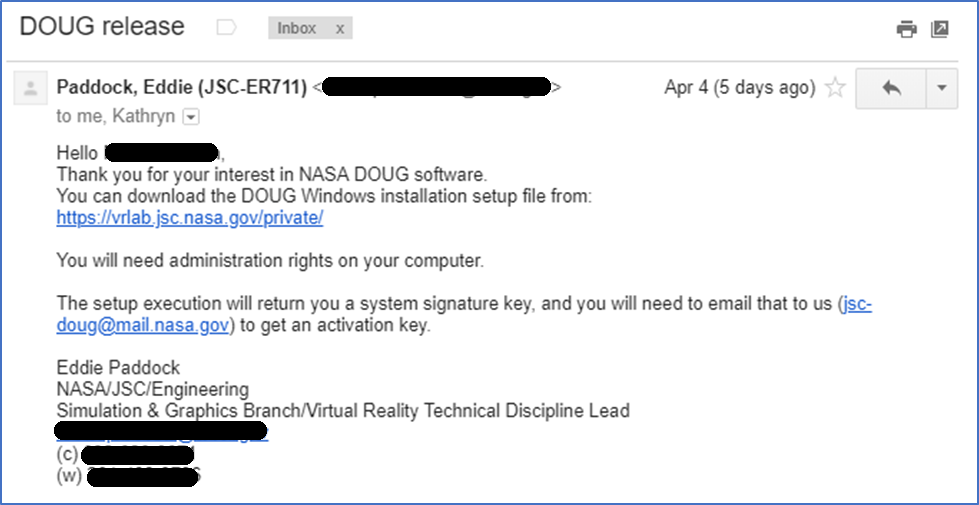


### Step 3: Approval Confirmation

Once submitted, it will take time to receive access. For our phase, it took 14 days before receiving the confirmation emails below.





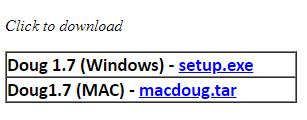


### Step 4: Download DOUG Installation

From the link in the DOUG Release, navigate to the DOUG Installation Setup File at

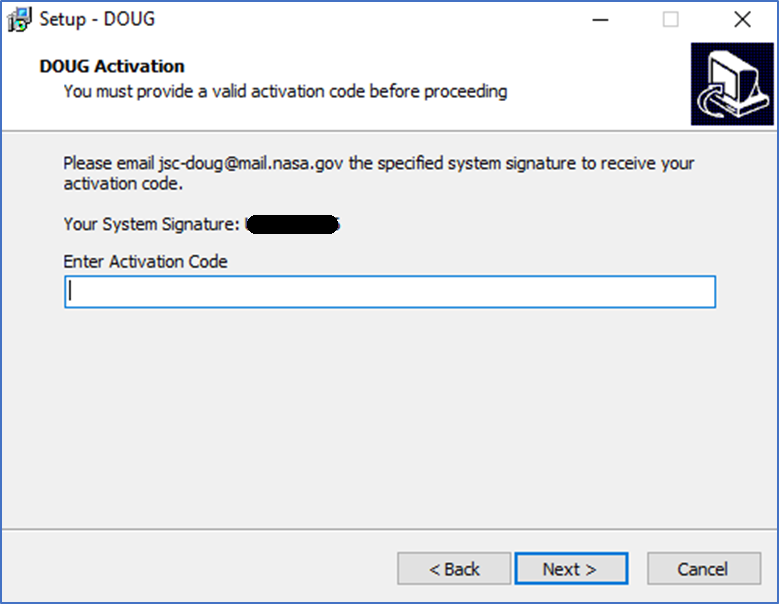
<https://vrlab.jsc.nasa.gov/private/>

Select respective OS:



Run respective installation setup file.

Copy the value of “Your System Signature: XXXXXXXXXX”



Email System Signature to jsc-doug@mail.nasa.gov to receive activation code.

### Step 5: Activation Code

After forwarding the System Signature, it will not take long to receive the activation code (less than 30 minutes for my request).



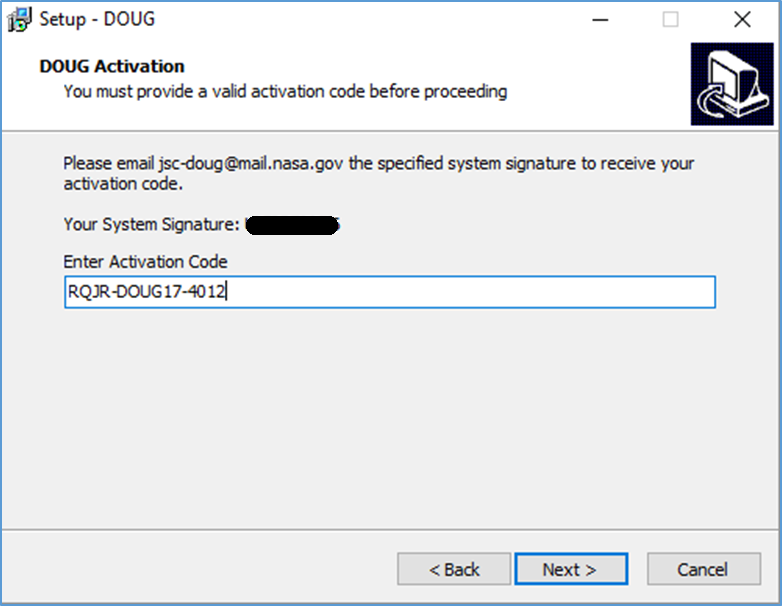
This Activation Code (Format: XXXX-XXXXXX-XXXX) will be used in the Installation process.

## Installation

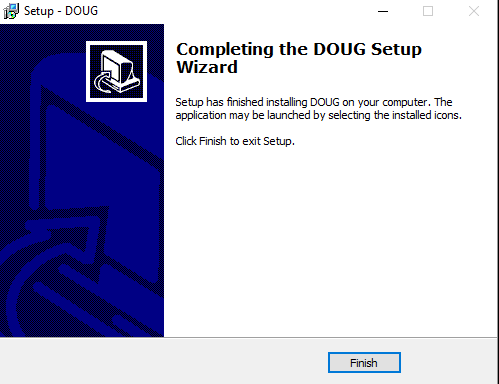
### Step 1: Run DOUG Setup Installation

After receiving the Activation Code, the DOUG software can be installed. Run the Installation Setup file downloaded in Registration Step 4.

### Step 2: Paste Activation Code

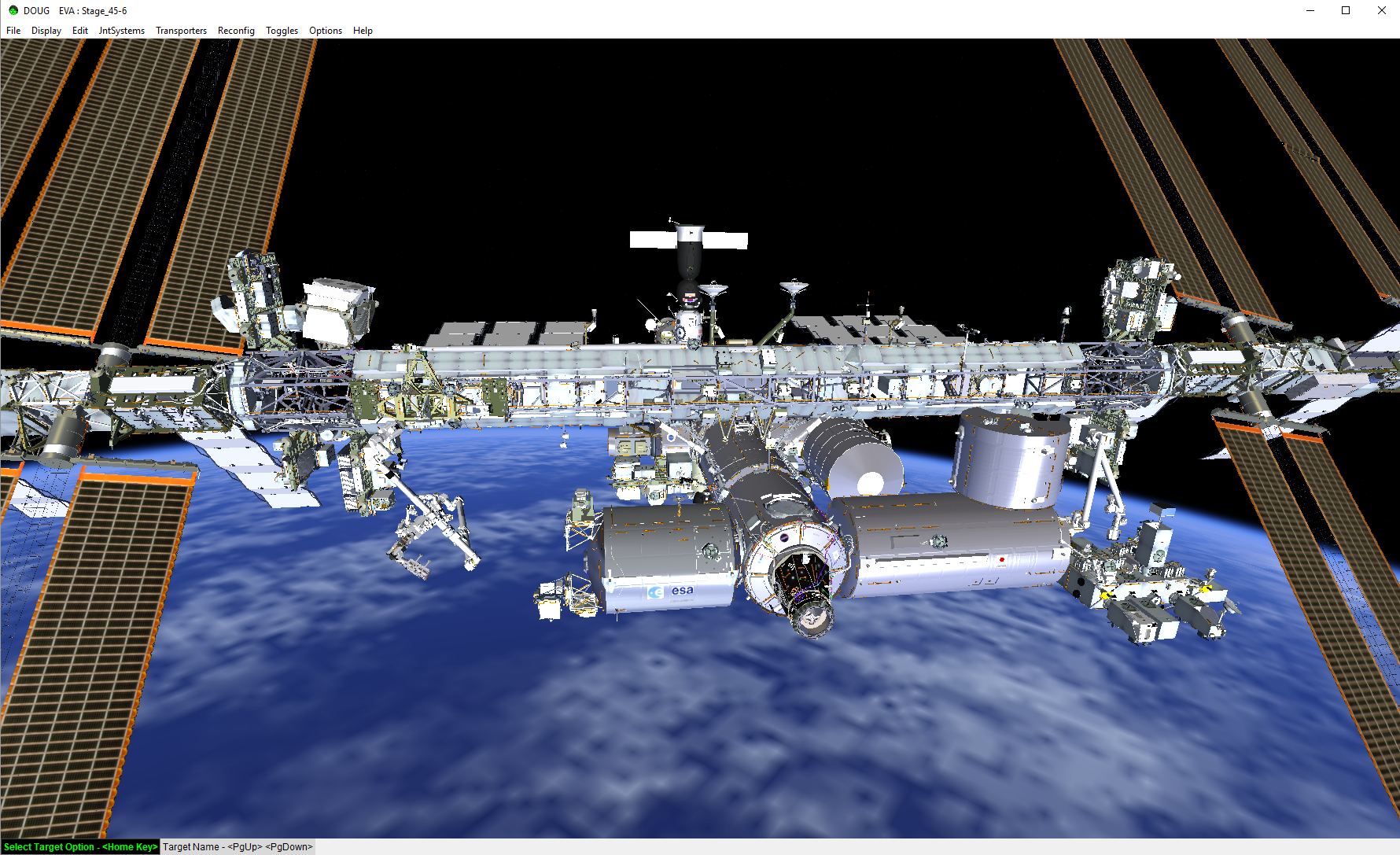


### Step 3: Complete Wizard



### Step 4: Confirm Installation

Installation complete! You should be able to click on the DOUG icon created during installation and wait for the application to render:



# Getting Started

## Launching the Application

To make the launch process simpler, you can use the script files to launch and run commands. Depending on your operating system you will want to select the corresponding script.



The first file above provides bash script for Ubuntu installations, the second provides batch script for Windows installations. Always refresh your browser once all terminal windows have fully loaded since the browser does not have the ability to determine if the back-end has loaded yet.

### Windows

To run the Windows batch script, double click the batch file (for easy access, you can make a shortcut on your desktop of this file).



Note: The launch files will open all the windows and execute the commands for you, but the browser does not query the dashboard to determine if the back end code has loaded. Always do a **browser refresh** once all terminals have completed their work and prior to use of the EVA Navigator.

Note 2: To turn on file extensions, open Control Panel > Appearance and Personalization. Click File Explorer Options > View tab. Uncheck Hide extensions for known file types.

### Ubuntu

To run the bash script in Ubuntu, cd into the nasa-path-finder and run the bash file.

* ./run\_nasa

Note: The launch files will open all the windows and execute the commands for you, but the browser does not query the dashboard to determine if the back end code has loaded. Always do a **browser refresh** once all terminals have completed their work and prior to use of the EVA Navigator.

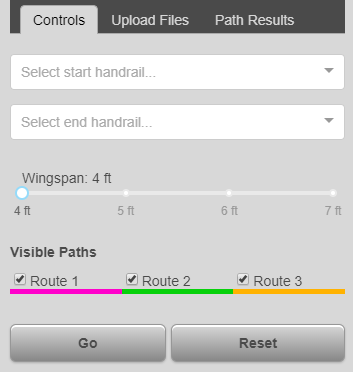
# Navigating the Navigator

The EVA Navigator application uses a simple UI that consists of a bar across the top with the NASA logo, three tabs on the left hand side that provide the user with the ability to control the navigator, and the main section that displays the model.

## Controls Tab

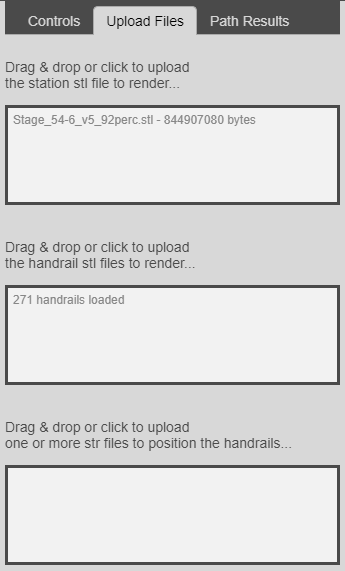
The controls tab allows users to select start and end handrails and their preferred wingspan in feet. The tab additionally provides the option to turn routes on and off in the model view.

Note: Some paths share nodes and will conceal each other when all three routes are selected.



## Upload Files Tab

The Upload Files tab provides three drag and drop fields that enable users to update the model and the handrails, as well as the handrail locations. The first and second fields accept stereolithography (STL) files. STL files are a 3D CAD format common in various modeling software and frequently used by 3D printers. Overly large STL files can crash your navigator, for more information read the Common Errors and Fixes section discussed below. The software determines the locations and positions of the handrails through the use of a structure list object file (STR) file. STR files contain a list of important information pertaining to each of the handrails.



The order of information in the STR follows this structure:

NodeName

ModelName

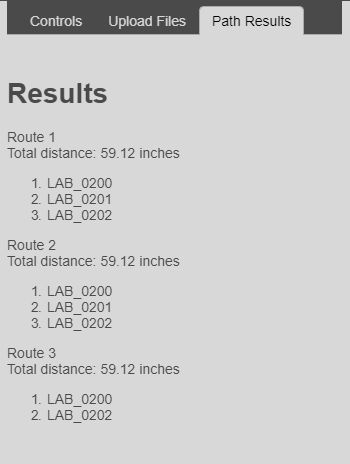
x y z

pitch yaw roll

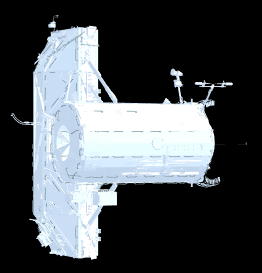
ParentNode

## Path Results Tab

The Path Results tab provides the route information for each of the three paths and provides the total path distance in inches.



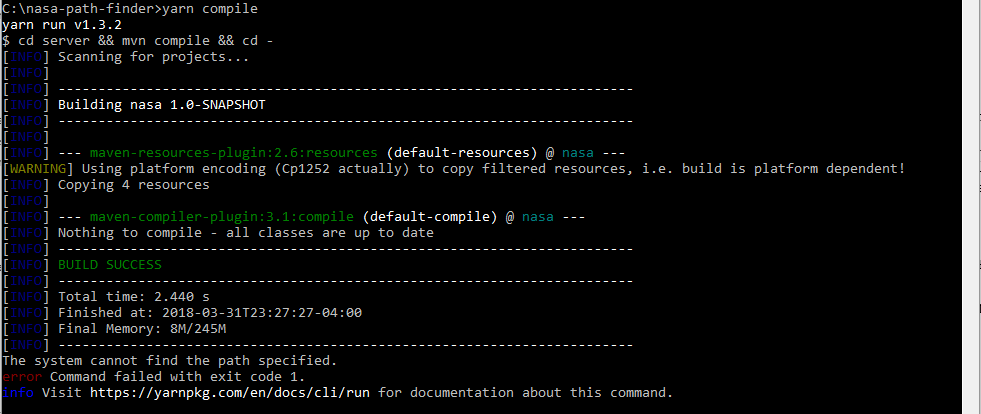
## Overall Model

The model pivots around a center point and clicking and dragging anywhere in the browser window will control the movement of the model. In future iterations, control will be added to move the camera’s pivot point dynamically to enable the user to select the area they wish to view.

# Common Errors and Fixes

## Compile Exit Code 1

If you are receiving the exit code 1 error, you are compiling for the wrong operating system. The Windows compilation can be run with “compilewin” or run with “compilewin:start:server”.



## Model Time Out

### Fix One:

Multiple reasons exist for the model to time out, one of them might be that Windows has added a safeguard to prevent your browser from taking too long. Since the models used by this software can sometimes take longer than the timeout allows, you will need to remove this attempt to block your browser from continuing.

1. Open Notepad as Administrator:

Right click on Notepad and select Run as Administrator

1. Browse for a file:

File > Open...

Select All Files (\*.\*) instead of the default Text Documents (\*.txt)

Browse to C:\Windows\System32\drivers\etc

Open the file named “hosts”

1. In the file:

Delete anything below the lines that start with #

### Fix Two:

Browsers have an internal timeout settings. You can look up the methods to change these in your preferred browser. As of the writing of this document, the most effective browser has been Chrome, however, Chrome does not allow users the option of changing the timeout settings and it limits us to a 92% version of the ISS model. Further research is needed.

### Fix Three (Pending):

In future iterations of the software an internal method will be added to prevent the page from getting caught up through the use of a loading manager. Three.js provides a LoadingManager method to handle and keep “track of loaded and pending data”. We strongly recommend the implementation of this method in conjunction with breaking up the model into smaller bits and loading each section separately.

# Packages and Libraries

If you wish to create a new dependency for the project, you must first create a package.json for the new library and then add it as an nmp install to the application. To do so, create a package and then install it as described below.

## Create a Package – package.json

If your package doesn’t have a package.json file, you can create one by doing the following after you’ve forked the git repo and cloned it onto your system.

1. In your command prompt cd into the directory you want to add the json file.

cd <directory path>

1. Type in the command to start the utility that walks you through creating the json file.

npm init

1. Hit enter to use the default package name, which is the directory name, or give it a new name if you’re so inclined.
2. Hit enter to keep the 1.0.0 version or type in number to change it.
3. Hit enter to leave the description blank or type one in to add it.
4. Type in the entry point for the package, typically index.js, if your package has one already you can once again just hit enter. (If your package doesn’t have one you might have to create one, this is the part I’m still unsure of as I’ve tried using a file that the package uses but that doesn’t seem to work.)
5. Test command: If your package has a test file, you can enter the name here, otherwise hit enter.
6. Git repository: Enter it here, otherwise hit enter.
7. Keywords: Type some in, or hit enter to continue.
8. Author: Name, if you want, email in angle brackets, website in parentheses if you have one.

Example: Your Name <you@email.com> (http://your.website.com)

1. Accept default license or type in your information.
2. Type yes to confirm your information was entered correctly.
3. To check that the package was created type dir to show the contents of the current directory, it should include package.json. Or you can just navigate to the folder and look inside it to see that you have one.

dir

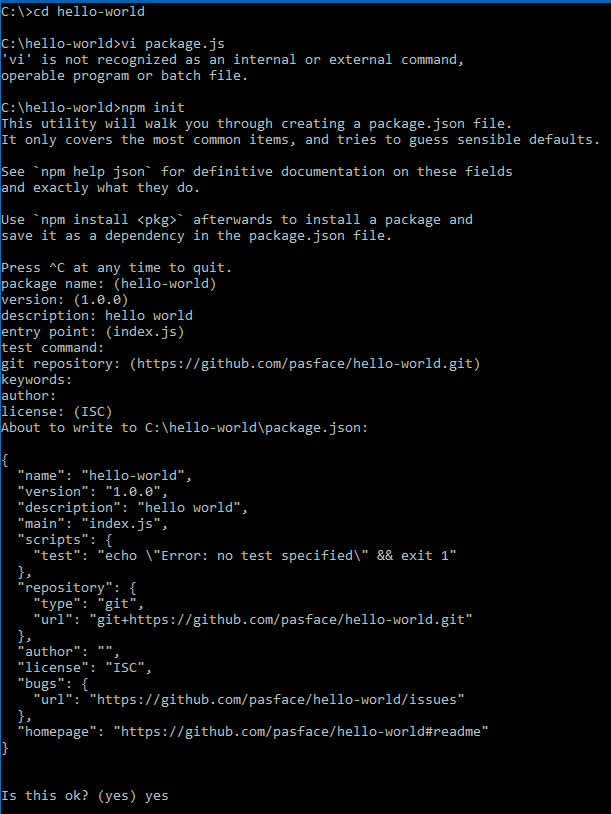
1. Push your update to your github fork.

git add .

git commit –m “added package.json”

git push

Screenshot in hello-world directory and keeping the defaults:



## Install NPM Package

To add a package to the application, you need to install it into the npm modules.

To install the package you need to cd into your application directory and type in npm install. This will add the package specified to the dependencies list in the application package.json and install the package into the node\_modules folder of our application.

cd ../nasa-path-finder

npm install <name of package>

Note:If the package you’re installing is one available through NPM, just use the name of the package and voila.

Note 2: If the package you’re trying to install is a GitHub repository, you will need to type in the Git location in the name of package section, but beware that if the package isn’t formatted correctly you may have to revert since it will break things.

# Appendix A: Glossary

DOUG – Dynamic Onboard Ubiquitous Graphics

EVA – Extra-Vehicular Activity

GPU – Graphics Processing Unit (Graphics Card)

ISS – International Space Station

SSD – Solid State Drive

SRS – Software Requirements Specification