CesiumJS: Intermediate Applications

The CesiumJS: Intermediate Applications course continues to build on the concepts learned in the Fundamentals path. By the end of these courses, developers should be able to create and run a web application that utilizes Cesium's higher-level functions.

Learning objectives:

- Understand and implement interactions
- Understand and implement general mapping concepts
- Set up and run a CesiumJS web application

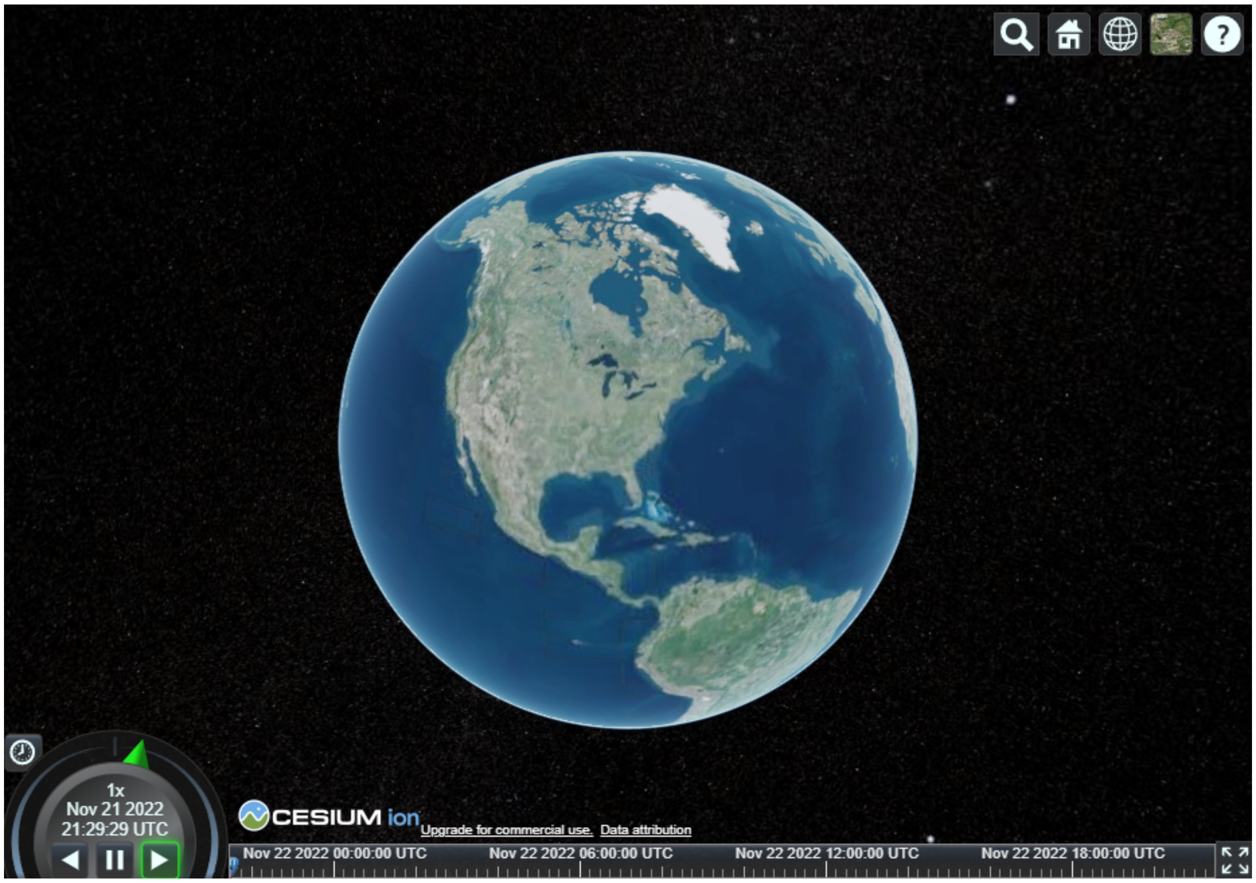
Lesson 1: Interactions in CesiumJS

Introduction:

This lesson will cover how to handle user interactions within your CesiumJS application.

Why are interactions important?

Most web applications will require user interaction, and CesiumJS comes with a few UI elements in the viewer that deal with the most common interactions. Still, there are times when your application will need additional support and customization. You may also need to pair down the default UI if the Cesium widget isn't needed, as shown in this sandcastle example.



Enabling interactions beyond the basic controls in the default viewer will be critical to developing an application that goes beyond a foundational level covered in the previous learning path. Let's look at how we can use the features CesiumJS provides to help deal with these scenarios.

How can I make the most of user interactions?

CesiumJS has a few built-in functions that help with capturing everyday interactions via the mouse and gestures. For example, `ScreenSpaceEventHandler` and `ScreenSpaceEventType` provide a way to listen to user inputs and categorize them. Developers can also incorporate native Javascript event listeners into their applications if the added complexity isn't required.

Assignment:

- Follow along with the [Drawing on 3D Models and Terrain] blog post describing how `Scene.pick` and `Scene.pickPosition` can enable dynamic user interactions.
- Recreate the Sandcastle example shown in the Drawing on 3D Models and Terrain blog post and rebind the event types to your preferred inputs.
    - Try using keystrokes or gestures and recreate the same functionality.

Additional Resources:

This section contains helpful links related to the lesson's content that might be interesting or helpful.

- If you want to see different ways to capture and utilize user interactions, look at the following Sandcastle examples:
    - Window Comparison
    - Moveable Cutout
    - Controlling a Model
- Cesium's reference documentation for `SceenSpaceEventType` and `ScreenSpaceEventHandler` cover the syntax supported.

Knowledge Check:

This section contains questions to check your understanding of this lesson. If you're having a problem answering a question, click the link and review the content again.

- Which events does `ScreenSpaceEventType` cover, and which are not?
- Which feature enables user interactions with specific features of 3D Tiles?

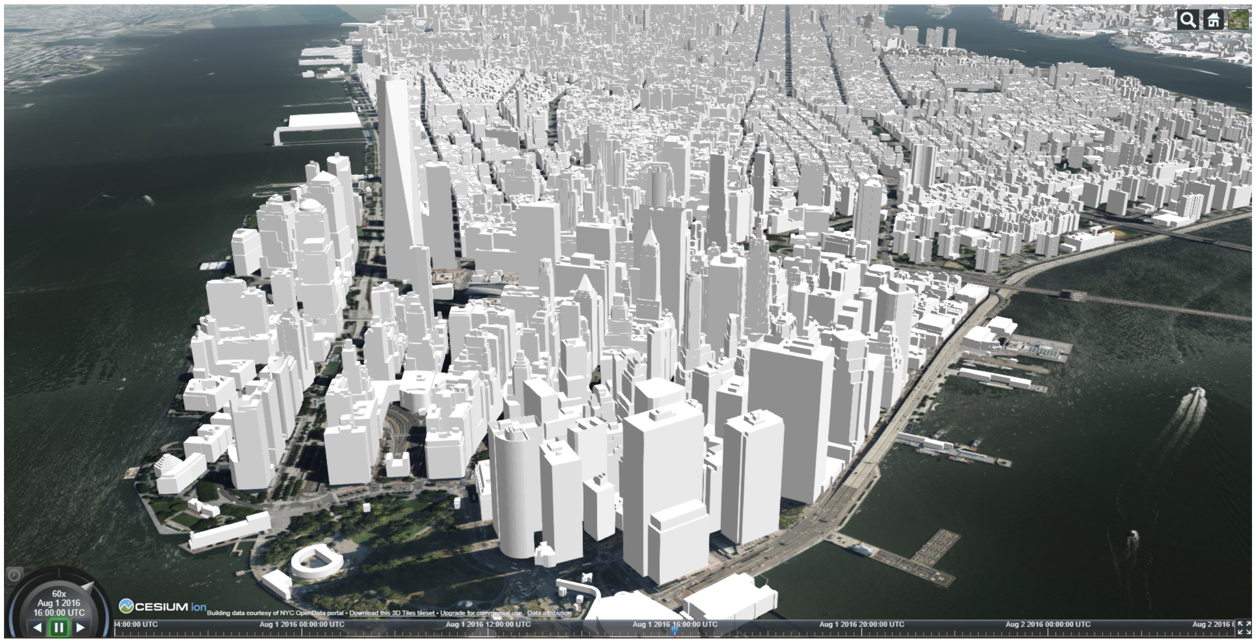
Lesson 2: Mapping Concepts in CesiumJS

Introduction:

This lesson will cover geospatial and mapping concepts important for creating a practical application with CesiumJS.

What is the difference between 2.5D and 3D?

Most people know the difference between 2D and 3D when it comes to maps. Think of the difference between a flat map you lay on your table versus a globe. The tricky part comes when trying to distinguish between 2.5D and 3D visualizations. Many 2D mapping tools attempt to bridge the gap between 2D and 3D, resulting in 2D maps with extrusions representing buildings and structures rather than 3D data.

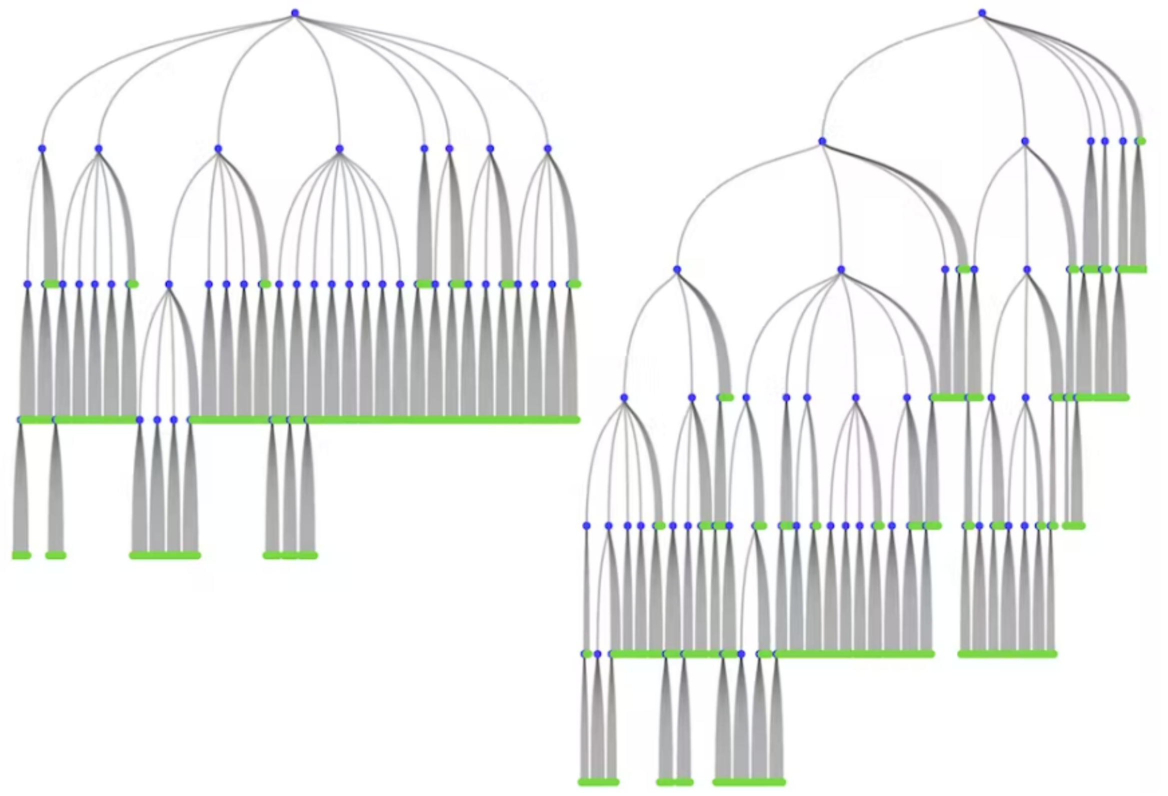


Unlike 2.5D representations, 3D engines capture the height detail shown in the image above. In a 2.5D map, the extruded footprint can only represent each building. As a result, accuracy in 2.5D falls behind that of 3D due to the limitations.

Why are data structures important in mapping?

We discussed some benefits of using 3D over 2D and 2.5D, but 3D data comes with some downsides, one being the size of the data. Cesium created 3D Tiles to deal with the size and complexity of heterogeneous data, and CesiumJS was built to stream and render those tiles as efficiently as possible.

Traditional geospatial applications utilize data structures such as quadtrees to manage the data efficiently, and 3D Tiles can still use that data structure, but 3D data is better suited to other optimized spatial subdivisions, such as octrees and k-d trees.


Left) Optimized non-uniform octree. Right) Traditional uniform octree.

Optimizing data structures, such as the left tree shown above, helps reduce streamed and stored data inefficiencies so that CesiumJS can operate on a browser without running into performance or memory problems.

What is the difference between vector and raster data?

If you are new to geospatial data, you may need to learn the difference between vector and raster data. Since both are common, it is helpful to understand how each type of data can be used in Cesium.

First, let's look at raster data to understand how it is used. Raster data is best described as a dataset made of pixels. Most people will associate raster data with images because they can contain visual information. Still, raster data can also be used to capture information, such as the height of the terrain or soil types.

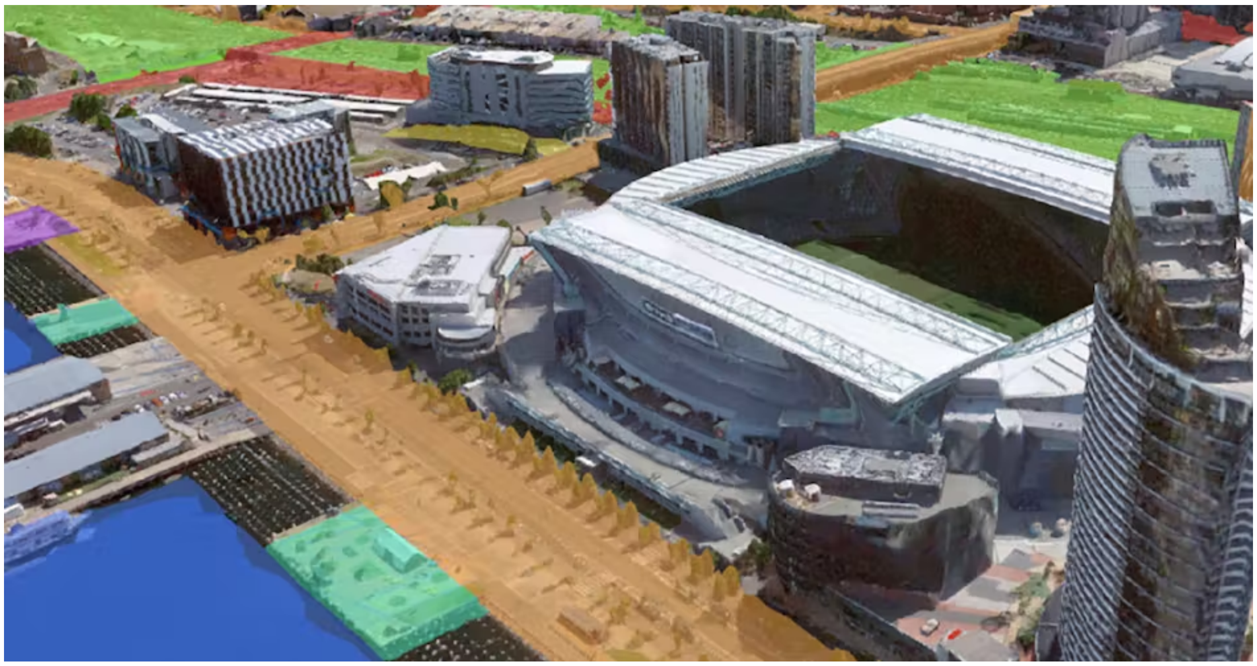
Vector data, on the other hand, is comprised of points, polylines, and polygons. Because of the networked structure of the data, vector datasets are often used to capture information about precise locations such as roadways and geographic boundaries.

Since Cesium can work with both types of data, it’s not required to use one or the other, but it helps to understand the benefits of each when creating practical geospatial applications.

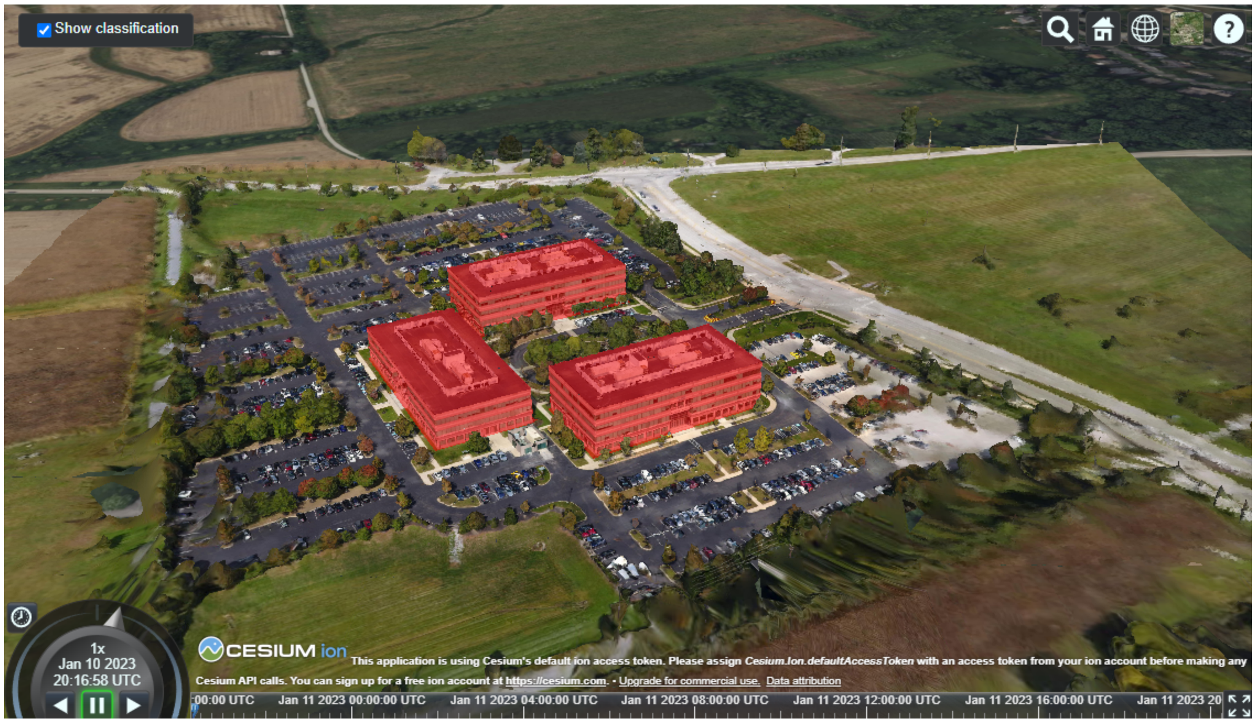
Why is data fusion important?

You may have come across the term “data fusion” in other contexts, but in the context of Cesium, we think of data fusion as showing multiple sources of data together in a combined manner so that we can understand how they intersect, overlap, and interact with each other. Multiple approaches in CesiumJS allow users to take advantage of the potential for data fusion, including classification, styling, and filtering.

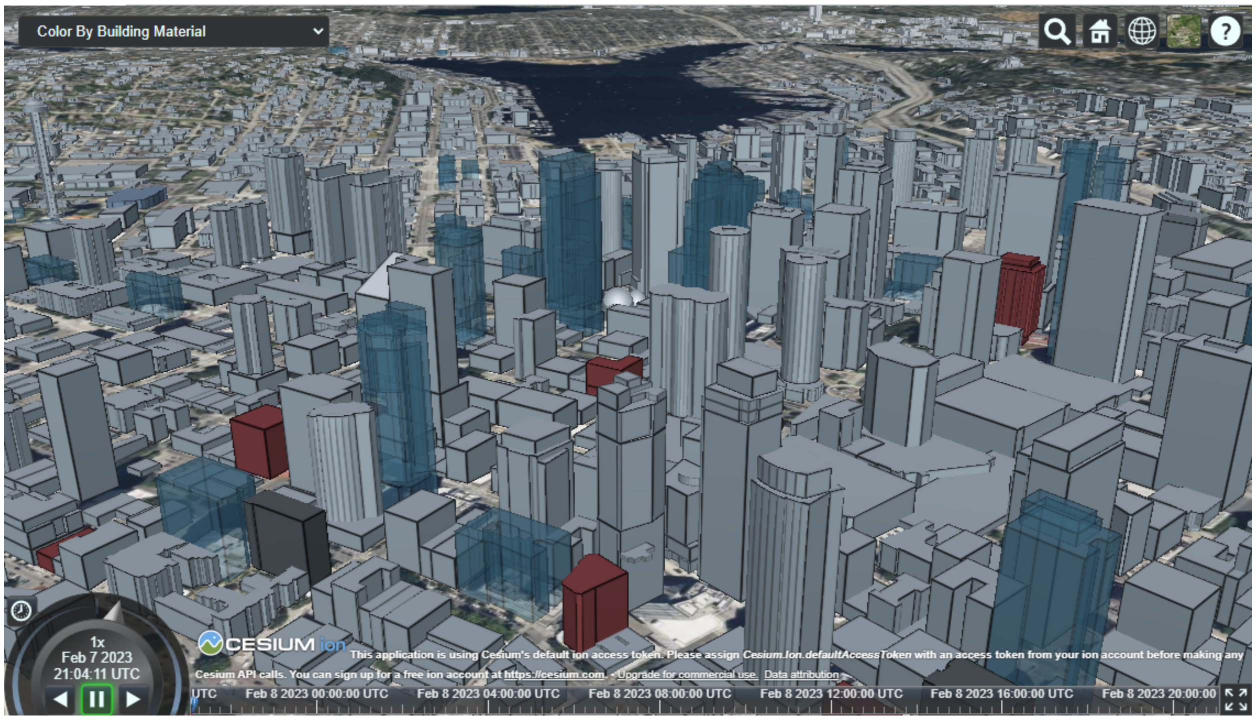
Classification is a flexible system in CesiumJS that allows you to highlight and annotate your data using geometries or tilesets. Classifications allow you to visually differentiate between features, even if they occur over different datasets which can be most helpful when you have metadata consistently across multiple tilesets.



Classifications can also differentiate between terrain and 3D tilesets, [as shown in this sandcastle example](), where the same geometry is used to classify either terrain, 3D tiles, or both simultaneously.


Show classification
1x
Jan 10 2023
20:16:58 UTC
CESIUM ion
This application is using Cesium's default ion access token. Please assign Cesium.Ion.defaultAccessToken with an access token from your ion account before making any Cesium API calls. You can sign up for a free ion account at https://cesium.com. • Upgrade for commercial use. Data attribution
Jan 11 2023 00:00:00 UTC
Jan 11 2023 04:00:00 UTC
Jan 11 2023 08:00:00 UTC
Jan 11 2023 12:00:00 UTC
Jan 11 2023 16:00:00 UTC

Styling allows for faster visual analysis of different data sets using shared metadata. The resulting patterns and intuitive visualizations provide users with an easy tool for highlighting the data that would otherwise be limited. Filtering takes the concept a little further by limiting which portions of the data are visible, which you can explore in this sandcastle example.


Color By Building Material
1x
Feb 7 2023
21:04:11 UTC
CESIUM ion
This application is using Cesium's default ion access token. Please assign Cesium.Ion.defaultAccessToken with an access token from your ion account before making any Cesium API calls. You can sign up for a free ion account at https://cesium.com. • Upgrade for commercial use. Data attribution
Feb 8 2023 00:00:00 UTC
Feb 8 2023 04:00:00 UTC
Feb 8 2023 08:00:00 UTC
Feb 8 2023 12:00:00 UTC
Feb 8 2023 16:00:00 UTC

Assignment:

- Read through Cesium's 2.5D is not 3D presentation to understand the underlying differences between both representations.
- Read through Optimizing Subdivisions in Spatial Data Structures and Optimizing Spatial Subdivisions in Practice.
- Follow along with the Dynamically Annotating 3D Tiles blog post to understand how classification can be quickly included.
- If you haven't gone through the tutorial yet, take some time to get familiar with the concepts covered in the Styling and Filtering 3D Tiles tutorial.

Additional Resources:

This section contains helpful links related to the lesson's content that might be interesting or helpful.

- This short article covers some basic concepts used in quadtree data structures.
- In this lesson, we discussed some of the downsides of 2D and 2.5D maps. If you want to learn more about 2D map projections, read this article by GISGeography.
- Another article by GISGeography covers vector and raster data in more detail.
- Explore the Photogrammetry Classification sandcastle example to see how highlighting can be toggled on/off per tileset.
- Explore the Data Fusion with 3D Tiles presentation for more context on the topic.

Knowledge Check:

This section contains questions to check your understanding of this lesson. If you're having a problem answering a question, click the link and review the content again.

- In which ways do 2.5D and 3D maps differ?
- What are some of the pros and cons of 2D geospatial applications?
- What are some of the pros and cons of 3D geospatial applications?
- Why is it important to have an optimized data structure for storing 3D Tiles?
- What is a practical example of using vector data?
- Why is classification through geometry used?

Lesson 3: Running a CesiumJS application

Introduction:

This lesson will cover the concepts needed to build a sample web application using Webpack.

Why is module bundling important?

If you're not familiar with them, module bundlers are tools that bundle all of your application's JavaScript modules into a single file that a browser can execute. Bundlers also help manage dependencies within your application and ensure that everything is loaded in the correct order.

In this lesson, we will focus on Webpack, which is just one of many module bundlers that developers can use to create web applications. Still, Rollup, FuseBox, and Parcel are all examples of module bundlers that you could also use.

How do I get started with Webpack?

If you have never used Webpack before, don't worry. We have a detailed guide on GitHub that walks you through the steps to get your CesiumJS application running locally in your browser through Webpack. If you have completed the previous lessons in the CesiumJS learning path then you should be ready to take on the tutorial in the Assignment section at the end of this lesson.

What should I do if I need to run CesiumJS offline?

If your application is not able to be connected to the internet or needs to operate with limited connectivity, there are still options for using CesiumJS. In the Assignment section, we will walk through the steps required to disconnect CesiumJS from external sources

Assignment:

Assign the learner a task to further their understanding of the topics covered in this lesson. The assignment could be an external reading, some targeted research, or creating a small application.

- Follow along with the CesiumJS and Webpack tutorial on GitHub. Once you are finished, you should be able to successfully run your own Cesium web application locally, which you will need for the practical application.
- Read through the GitHub documentation for setting up CesiumJS in an offline environment. Even if your application is online, it's helpful to understand which components require connectivity.

Additional Resources:

This section contains helpful links related to the lesson's content that might be interesting or helpful.

- You should have already completed the CesiumJS Quickstart tutorial if you went through the CesiumJS: Fundamentals learning path, but if you haven't, go through the tutorial now.
- If you are interested in exploring on-prem solutions take a few minutes to read the Cesium On-Premise page.
- Webpack offers a Getting Started tutorial that goes a little more in depth on the specifics of bundling in Webpack.

Knowledge Check:

This section contains questions to check your understanding of this lesson. If you're having a problem answering a question, click the link and review the content again.

- What are the benefits of using a bundler when creating a web application?
- What are some examples of static assets that will need to be served alongside the CesiumJS javascript files?
- Which two widgets should be disabled for CesiumJS to work in an offline environment?

Practical Application

Now that you understand the concepts needed for intermediate applications in CesiumJS, you need to put them into practice. Before progressing onto the next Learning Path, create a CesiumJS web application that utilizes the following concepts:

1. Create at least two different types of user interactions
2. Demonstrate how geometry can classify terrain or 3D tilesets
3. Bundle your web application using Webpack

Getting Certified:

At this point, you have learned all of the concepts needed to become a Cesium Certified Developer. Now we want to see how you can apply these skills to a real problem. To receive certification for your work, your application must also meet these criteria:

1. Demonstrate how your application solves a real-world use case.

2. Provide explanations on how you pinpointed this use case and why Cesium works well to solve it.
3. Describe any additional integrations used that improved your application, i.e., 3rd party data, other JS libraries, etc.
4. Present your application and code for review.