SnapshotTestRunner

Client-side utility for browser-based deck.gl render tests.

This class is intended to be used with `BrowserTestDriver` from `@probe.gl/test-utils`. Together they support the following workflow:

- Launch a Puppeteer instance (headless or non-headless) to run a test application
- In the test application, create a deck.gl canvas.
- For each test case, render a set of deck props including views and layers, take a screenshot, and perform pixel-diffing with a pre-defined "golden image". Report the matching result.
- Proceed to the next test case until done.

Example

In your node.js start script:

```
// This is the script that runs in Node.js and starts the browser
const {BrowserTestDriver} = require('@probe.gl/test-utils');
new BrowserTestDriver().run({
  server: {
    // Bundles and serves the browser script
    command: 'webpack-dev-server',
    arguments: ['--env.render-test']
  },
  headless: true
});
```

In your script that is run on the browser:

```
const {SnapshotTestRunner} = require('@deck.gl/test-utils');
const {ScatterplotLayer} = require('@deck.gl/layers');

const TEST_CASES = [
  {
    name: 'ScatterplotLayer',
    // `Deck` props
    viewState: {
      longitude: -122.4,
      latitude: 37.8,
      zoom: 12,
      pitch: 20
    },
    layers: [
      new ScatterplotLayer({
        id: 'circles',
        data: './data/scatterplot.json',
        getPosition: d => d.position,
        getRadius: d => d.size,
        getFillColor: [255, 0, 0]
      })
    ],
    // `done` must be called when ready for screenshot and compare
    onAfterRender: ({layers, done}) => {
      if (layers[0].props.data.length) {
        // data is loaded
        done();
      }
    },
    // Target rendering result
    goldenImage: './test/render/golden-images/scatterplot.png'
  }
];

new TestRender({width: 800, height: 600})
  .add(TEST_CASES)
  .run({
    onTestFail: window.browserTestDriver_fail
  })
  .then(window.browserTestDriver_finish);
```

Methods

constructor(props: Object)

```
new SnapshotTestRunner(deckProps)
```

Create a SnapshotTestRunner instance. The `deckProps` argument is passed to the Deck constructor.

add(testCase: Array|Object)

Add one or a list of test cases. Each test case may contain the following fields:

- `name` (String) - name of the test case.
- `goldenImage` (String) - path to the golden image, relative to the root where the node script is executed.
- `timeout` (Number) - time to wait for this test case to resolve (by calling the `done` callback) before aborting, in milliseconds. If not provided, fallback to the shared option that is passed to `SnapshotTestRunner.run`.
- `imageDiffOptions` (Object, optional) - image diffing options for this test case. See "Image Diff Options" section below.
- `onBeforeRender` (Function, optional) - callback before each time deck rerenders. Receives the following arguments:
  - `deck` (Deck) - the `Deck` instance.
  - `layers` (Array) - the list of layers that were rendered.
- `onAfterRender` (Function, optional) - callback after each time deck rerenders. Receives the following arguments:
  - `deck` (Deck) - the `Deck` instance.
  - `layers` (Array) - the list of layers that were rendered.
  - `done` (Function) - must be called when the test case is done rendering and ready for screen capture and comparison.

  The default `onAfterRender` calls `done` immediately, i.e. takes screenshot as soon as the canvas is rendered for the first time. If some resources are loaded asynchronously, you may need to provide an implementation of this callback to check whether all layers are fully loaded.
- Any other props that Deck.setProps accepts.

run(options: Object)

Run all test cases.

Options:

- `timeout` (Number) - time to wait for each test case to resolve (by calling the `done` callback) before aborting, in milliseconds. Default `2000`.
- `imageDiffOptions` (Object) - image diffing options for all test cases. This will be overridden if a test case defines its own `imageDiffOptions`. See "Image Diff Options" section below.
- `onTestStart` (Function) - callback when a test starts. Receives the current test case. Default logs the test name to console.
- `onTestPass` (Function) - callback when a test passes. Receives the current test case and the diffing result. Default logs the pixel matching percentage to console.
- `onTestFail` (Function) - callback when a test fails, either because the matching rate is below threshold or a critical error. Receives the current test case. Default logs the error message or the pixel matching percentage to console.

Returns: a `Promise` that resolves when all test cases are done.

Members

isHeadless

Whether the test is being run in headless mode. In headless mode, Chromium uses software render which behaves slightly differently from non-headless. Image diffing tolerance may need to be adjusted accordingly.

Image Diff Options

The test renderer and each test case may choose to override the default image diffing options. The following options from captureAndDiffScreen are supported:

- `tolerance`
- `threshold`
- `includeAA`
- `includeEmpty`

- `createDiffImage`
- `saveOnFail`
- `saveAs`

Edit this page