Hamburg Coding School

Henning Muszynski

Session 5

JavaScript for Web

Feedback

What did we learn last time

Re-examination

Web development in a nutshell

```
// create element
const linkElement = document.createElement('a')

// modify element
linkElement.href = 'https://google.com'
linkElement.innerText = 'Google'

// insert element
document.body.appendChild(linkElement)
```

Store data in LocalStorage

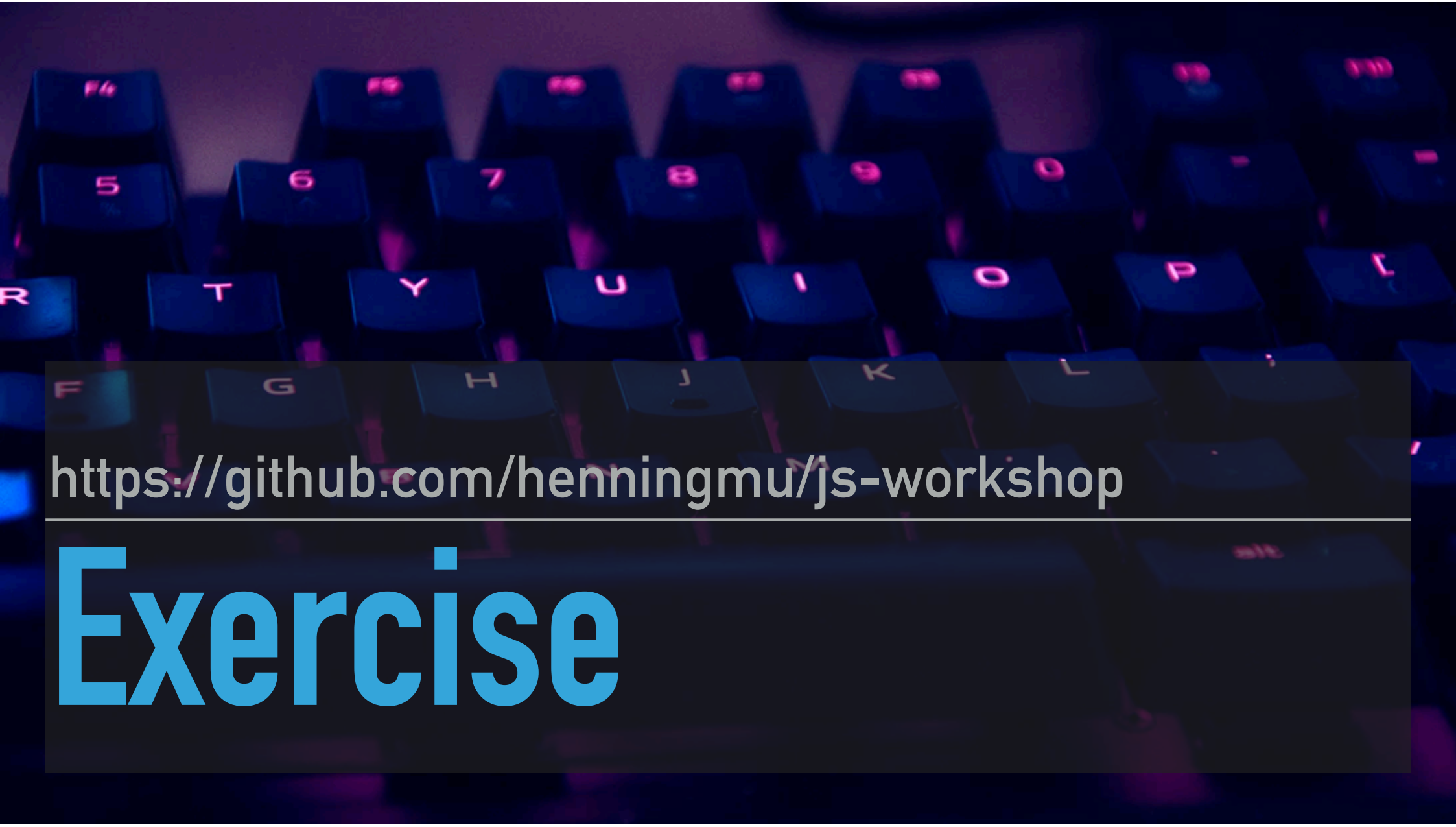
- Synchronous Key-Value Storage
- Only Strings as values

```
localStorage.setItem('key', value)

localStorage.getItem(‘key')

localStorage.removeItem('key')
```

Store data in LocalStorage

Read and write complex data

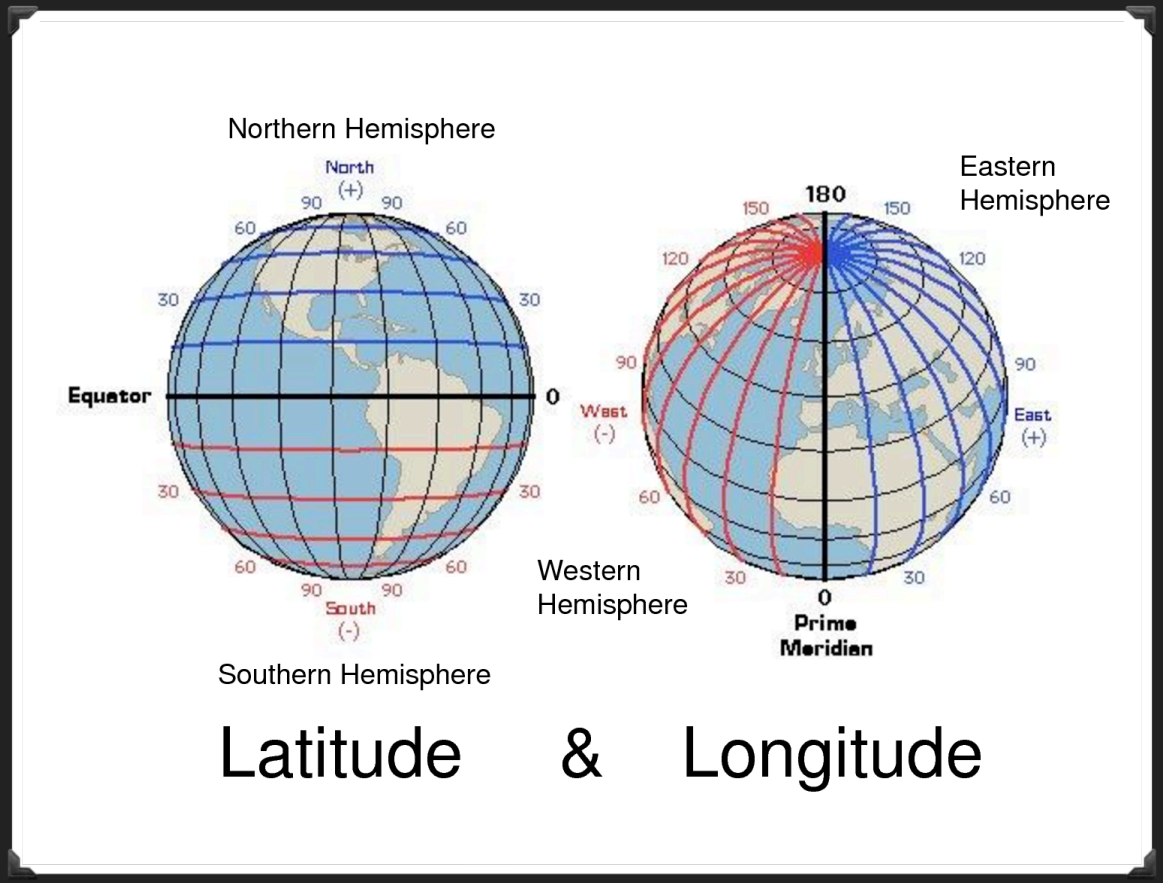
```
const data = [{ id: 1, text: 'Milk' }, { id: 2, text: 'Bread' }]

setItem('data', JSON.stringify(data))

JSON.parse(getItem('data'))
```



https://github.com/henningmu/js-workshop

Exercise

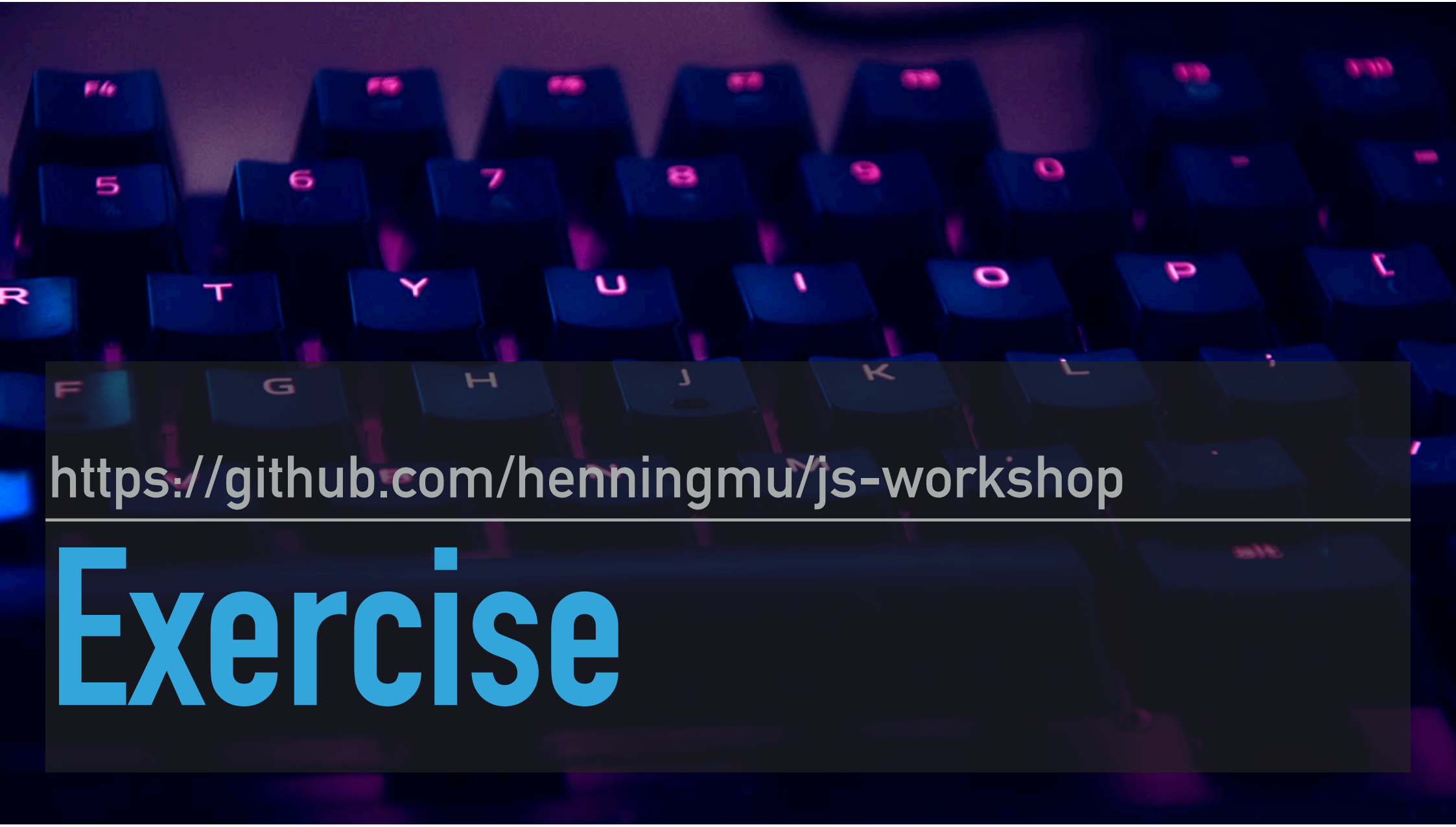
Get the current location with the Geolocation API



Get the current location with the Geolocation API

Requires user permission – so we should always handle errors gracefully

```
if (navigator.geolocation) {
    navigator.geolocation.getCurrentPosition(
        (position) => {
            const { latitude, longitude } = position.coords
            console.log(‘Location:', latitude, longitude)
        },
        (error) => {
            console.log('Oh no:', error)
        }
    )
}
```

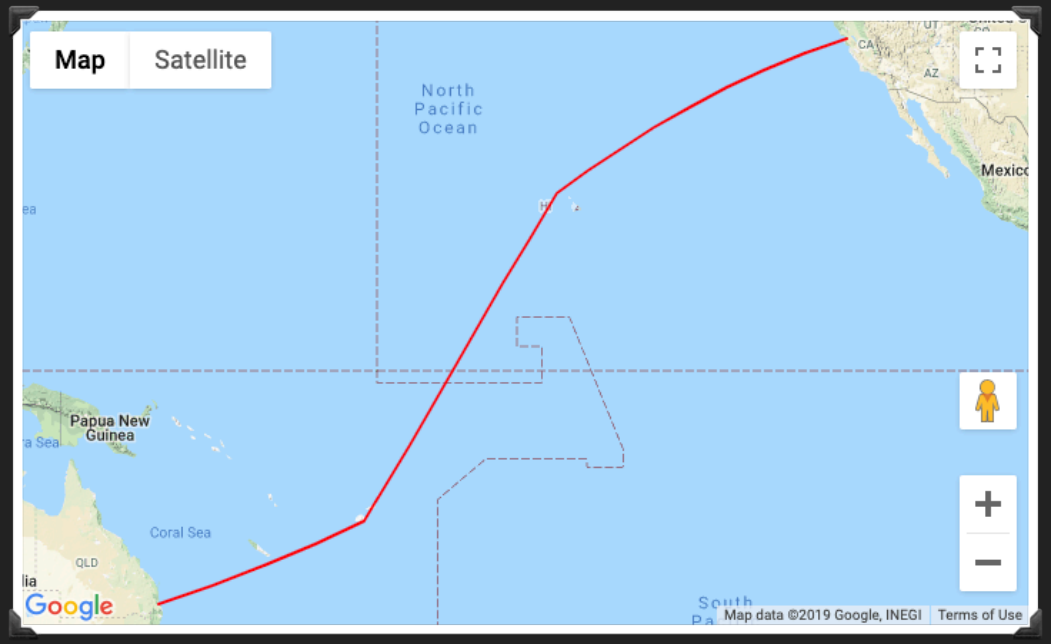


https://github.com/henningmu/js-workshop

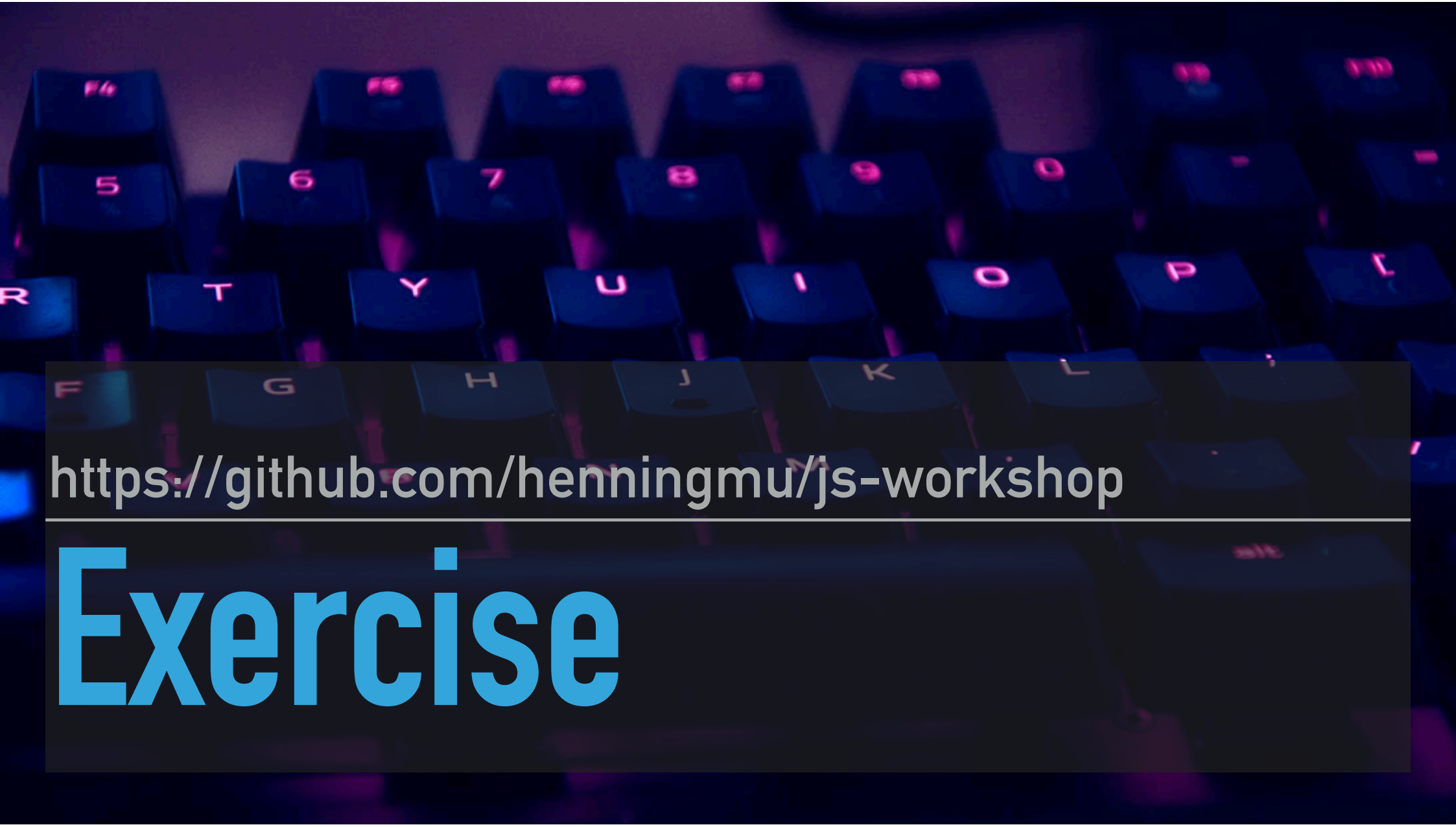
Exercise

Display a map with Google Maps API

- Interactive map of any location
- Zoom levels (continents to houses)
- Markers and Lines



```
<html lang="en">
  <head>
    <script src="googleapis.com?callback=initMap" async defer></script>
    <script>
      let map

      function initMap() {
        const mapElement = document.querySelector('#map')
        map = new google.maps.Map(mapElement, {
          center: { lat: -33.9249, lng: 18.4241 },
          zoom: 12
        })
      }
    </script>
  </head>
  <body>
    <div id="map"></div>
  </body>
</html>
```

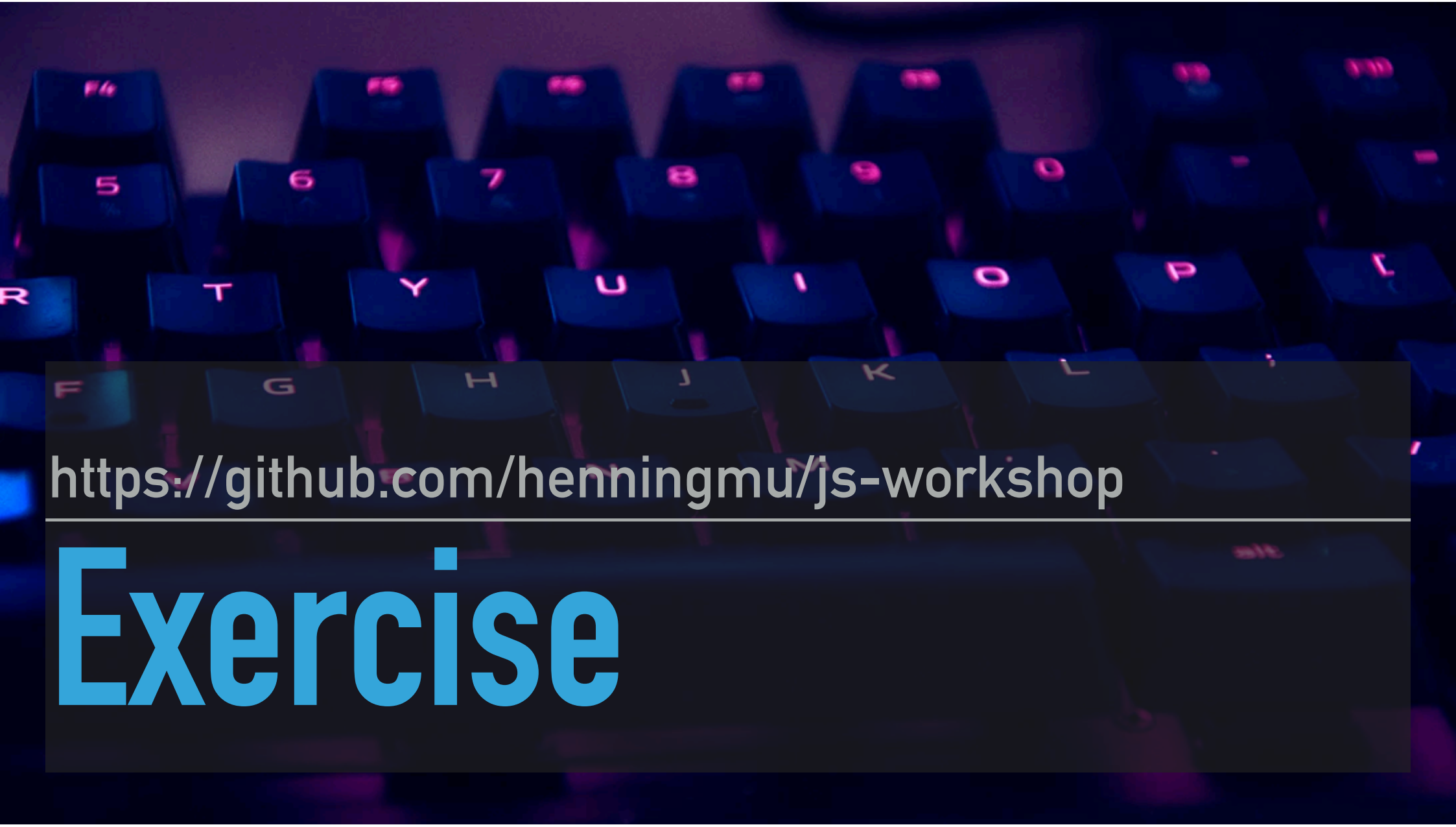


https://github.com/henningmu/js-workshop

Exercise

Showing a marker

```
const marker = new google.maps.Marker({
    position: { lat: -33.0249, lng: 18.4241 },
    map: map
})

// read position of marker
const position = marker.getPosition()
```

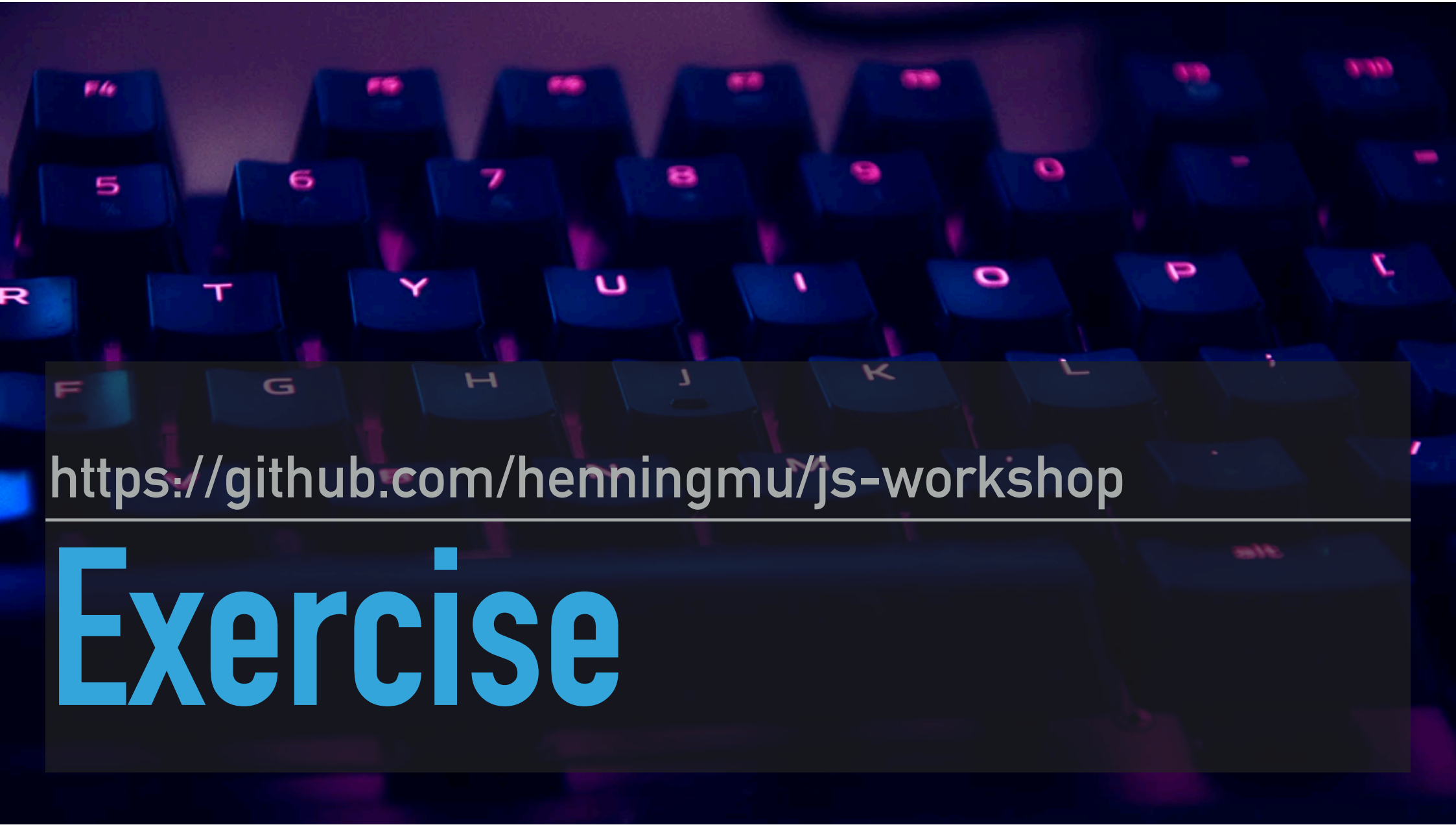


https://github.com/henningmu/js-workshop

Exercise

Handling bounds of the map

```
const bounds = new google.maps.LatLngBounds()
cities.forEach(city => {
    const marker = new google.maps.Marker(/* ... */)
    bounds.extend(marker.getPosition())
})
map.fitBounds(bounds)
```



https://github.com/henningmu/js-workshop

Exercise

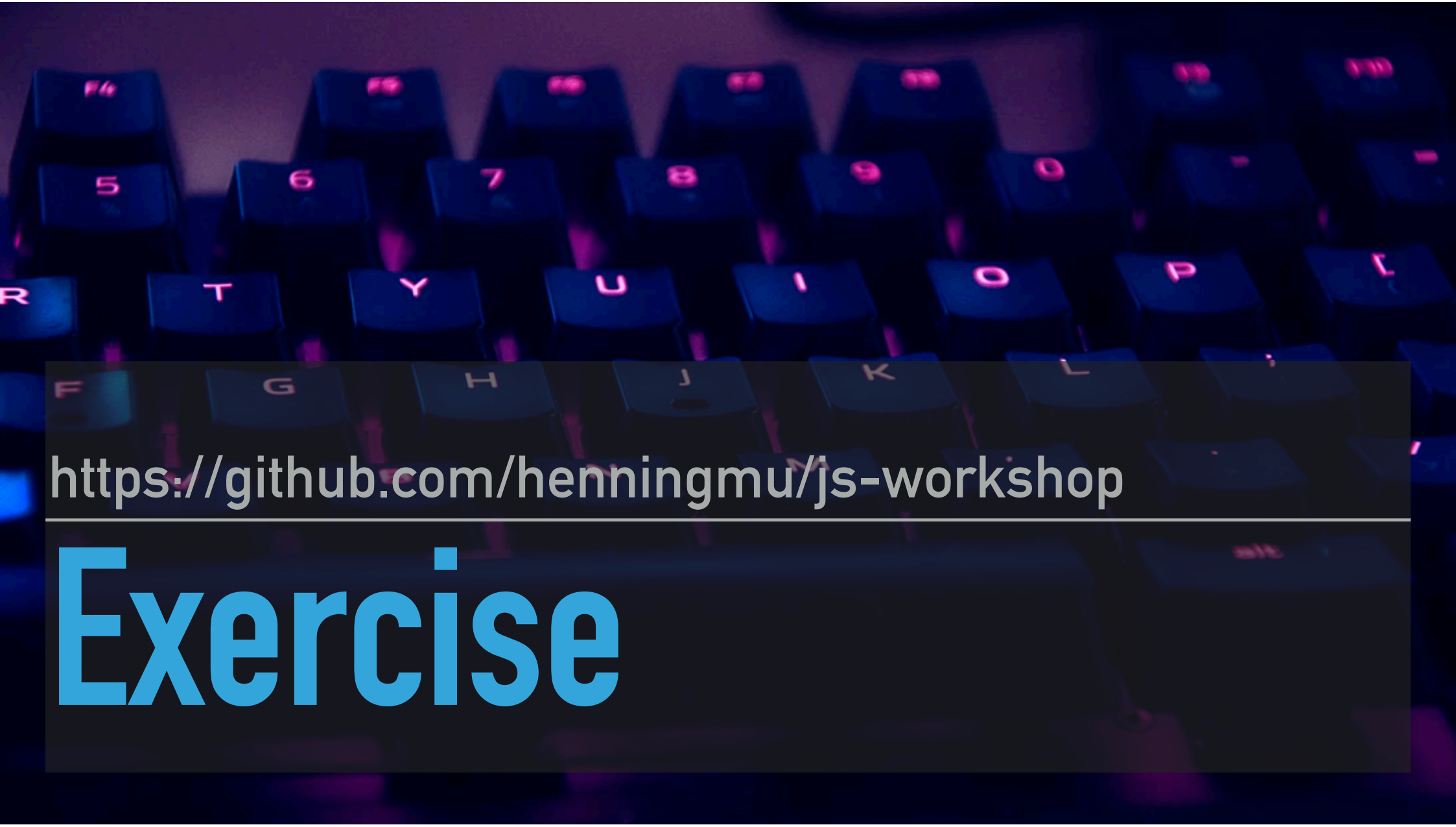
Handling events

```
map.addListener('click', event => {
    /* ... */
})
```

- Possible events include:
  - `click, dblclick, rightclick, mouseout, dragend`
  - `center_changed, bounds_changed`

Drawing a line

```
const line = new google.maps.Polyline({
    map: map,
    path: [
        { lat: -33.0249, lng: 18.4241 },
        { lat: -32.0249, lng: 19.4241 }
    ]
})
```



https://github.com/henningmu/js-workshop

Exercise

Reverse Geocoding

- New API that needs to be enabled in the Google Cloud Console
- Allows to look up addresses based on coordinates

Reverse Geocoding

```
const geocoder = new google.maps.Geocoder()
geocoder.geocode(
    { location: { lat: -33, lng: 18 } },
    (results, status) => {
        if (status === 'OK') {
            if (results) {
                console.log(results)
            } else {
                console.error('No results found')
            }
        } else {
            console.error('Geocoder failed due to: ' + status)
        }
    })
```

Quick Feedback