Notes

Firebase Storage

See the Firebase Storage docs for web.

Create a ref

```
var storageRef = firebase.storage().ref();
const fileRef = storageRef.child('/some/file/path.jpg);
```

Navigate

```
// Points to the root reference
var storageRef = firebase.storage().ref();

// Points to 'images'
var imagesRef = storageRef.child('images');

// Points to 'images/space.jpg'
// Note that you can use variables to create child values
var fileName = 'space.jpg';
var spaceRef = imagesRef.child(fileName);

// File path is 'images/space.jpg'
var path = spaceRef.fullPath;

// File name is 'space.jpg'
var name = spaceRef.name;

// Points to 'images'
var imagesRef = spaceRef.parent;
```

Upload file

```
// Create file metadata including the content type
var metadata = {
  contentType: 'image/jpeg',
};

// Upload the file and metadata
var uploadTask = storageRef.child('images/mountains.jpg').put(file, metadata);
```

Full example

```
function uploadFile(file) {
  // Create the file metadata
  var metadata = {
    contentType: 'image/jpeg',
  };

  // Upload file and metadata to the object 'images/mountains.jpg'
  var uploadTask = storageRef.child('images/' + file.name).put(file, metadata);

  // Listen for state changes, errors, and completion of the upload.
  uploadTask.on(
    firebase.storage.TaskEvent.STATE_CHANGED, // or 'state_changed'
    function(snapshot) {
      // Get task progress, including the number of bytes uploaded and the total
number of bytes to be uploaded
      var progress = snapshot.bytesTransferred / snapshot.totalBytes * 100;
      console.log('Upload is ' + progress + '% done');
      switch (snapshot.state) {
        case firebase.storage.TaskState.PAUSED: // or 'paused'
          console.log('Upload is paused');
          break;
        case firebase.storage.TaskState.RUNNING: // or 'running'
          console.log('Upload is running');
          break;
      }
    },
    function(error) {
      // Errors list: https://firebase.google.com/docs/storage/web/handle-errors
      switch (error.code) {
        case 'storage/unauthorized':
          // User doesn't have permission to access the object
          break;

        case 'storage/canceled':
          // User canceled the upload
          break;

        case 'storage/unknown':
          // Unknown error occurred, inspect error.serverResponse
          break;
      }
    },
    function() {
      // Upload completed successfully, now we can get the download URL
      var downloadURL = uploadTask.snapshot.downloadURL;
    }
  );
}
```

Download file

```
// Create a reference to the file we want to download
var starsRef = storageRef.child('images/stars.jpg');

// Get the download URL
starsRef.getDownloadURL().then(function(url) {
  // Insert url into an <img> tag to "download"
}).catch(function(error) {

  // A full list of error codes is available at
  // https://firebase.google.com/docs/storage/web/handle-errors
  switch (error.code) {
    case 'storage/object_not_found':
      // File doesn't exist
      break;

    case 'storage/unauthorized':
      // User doesn't have permission to access the object
      break;

    case 'storage/canceled':
      // User canceled the upload
      break;

    ...

    case 'storage/unknown':
      // Unknown error occurred, inspect the server response
      break;
  }
});
```

Set metadata

```
// Create a reference to the file whose metadata we want to change
var forestRef = storageRef.child('images/forest.jpg');

// Create file metadata to update
var newMetadata = {
  cacheControl: 'public,max-age=300',
  contentType: 'image/jpeg',
  contentLanguage: null,
  customMetadata: {
    whatever: 'we feel like',
  },
};

// Update metadata properties
forestRef
  .updateMetadata(newMetadata)
  .then(function(metadata) {
    // Updated metadata for 'images/forest.jpg' is returned in the Promise
  })
  .catch(function(error) {
    // Uh-oh, an error occurred!
  });
```