**Local Storage**

In order to persist data in a web application, we'd normally store the data in some sort of database. This prevents app data from being lost between page refreshes. Using localStorage, we can achieve a similar effect for the user by storing this data *directly in their browser*. Best of all -- data stored in localStorage has no expiration date. This means that even if a session ends (e.g. the browser tab is closed), data will not be lost!

Feel free to check out [**Window.localStorage**](https://developer.mozilla.org/en-US/docs/Web/API/Window/localStorage) on MDN for an overview.

**Example: Saving to localStorage**

Let's say we're building a simple React and Redux application that lets users create and manage a list of tasks. Basic functionality allows users to add items to their task list, remove items, and mark items as completed.

Assuming much of this data is kept in the application's store, how would we go about persisting this data? One way would be to save to localStorage each time that state is updated. That is, the store's state will be saved with each *dispatch*:

*// store.js*

**import** { createStore } **from** 'redux';

**import** Reducer **from** '../reducers/reducer';

**const** configureStore = () => {

**const** store = createStore(Reducer);

store.subscribe(() => {

localStorage.state = JSON.stringify(store.getState());

});

**return** store;

};

**export** **default** configureStore;

After the store is created, we call store.subscribe() and pass in a callback function. The callback effectively saves a JSON string of the store's state into localStorage. As a result, by subscribing to the store right after it is created, we can save data related to all of the user's tasks right into their browser!

<https://youtu.be/uO2dR3LPOs0>

The React Native documentation on [**AsyncStorage**](https://facebook.github.io/react-native/docs/asyncstorage.html) mentions:

AsyncStorage is a simple, unencrypted, asynchronous, persistent, key-value storage system that is global to the app. It should be used instead of LocalStorage.

In the next video, we'll see just how we can implement it into our app!

<https://youtu.be/243xzJEz7xo>

## Summary

React Native's version of localStorage is AsyncStorage. Conveniently, since AsyncStorage is just an abstraction over iOS and Android equivalents, there's no need to consider the different environments.

We took a close look at these 3 methods available on AsyncStorage:

* setItem
* mergeItem
* getItem

Feel free to visit the [**documentation**](https://facebook.github.io/react-native/docs/asyncstorage.html#methods) for a complete list.

#### Lesson Challenge

Answer the following questions and share your answers with your classmates:

1) Explain how AsyncStorage works.

2) Implement what you've learned about AsyncStorage to store/access items of your choosing. Implement the functionality of retrieving all of the items, retrieving a particular item by id, saving a new item.