```
/**
 * Blockly Games: Storage
 *
 * Copyright 2012 Google Inc.
 * https://github.com/google/blockly-games
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *   http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */

/**
 * @fileoverview Loading and saving blocks with localStorage and cloud storage.
 * @author q.neutron@gmail.com (Quynh Neutron)
 */
'use strict';

// Create a namespace.
var BlocklyStorage = {};

/**
 * Backup code blocks or JavaScript to localStorage.
 * @private
 */
BlocklyStorage.backupBlocks_ = function() {
  if ('localStorage' in window) {
    var code = BlocklyInterface.getCode();
    // Gets the current URL, not including the hash.
    var url = window.location.href.split('#')[0];
    window.localStorage.setItem(url, code);
  }
};

/**
 * Bind the localStorage backup function to the unload event.
 */
BlocklyStorage.backupOnUnload = function() {
  window.addEventListener('unload', BlocklyStorage.backupBlocks_, false);
};

/**
 * Restore code blocks or JavaScript from localStorage.
 */
BlocklyStorage.restoreBlocks = function() {
  var url = window.location.href.split('#')[0];
  if ('localStorage' in window && window.localStorage[url]) {
    var code = window.localStorage[url];
    BlocklyInterface.setCode(code);
  }
};

/**
 * Save blocks or JavaScript to database and return a link containing the key.
 */
BlocklyStorage.link = function() {
  var code = BlocklyInterface.getCode();
  BlocklyStorage.makeRequest_('/storage', 'xml', code);
};

/**
 * Retrieve XML text from database using given key.
```

```
 * @param {string} key Key to XML, obtained from href.
 */
BlocklyStorage.retrieveXml = function(key) {
  BlocklyStorage.makeRequest_('/storage', 'key', key);
};

/**
 * Global reference to current AJAX request.
 * @type XMLHttpRequest
 * @private
 */
BlocklyStorage.httpRequest_ = null;

/**
 * Fire a new AJAX request.
 * @param {string} url URL to fetch.
 * @param {string} name Name of parameter.
 * @param {string} content Content of parameter.
 * @private
 */
BlocklyStorage.makeRequest_ = function(url, name, content) {
  if (BlocklyStorage.httpRequest_) {
    // AJAX call is in-flight.
    BlocklyStorage.httpRequest_.abort();
  }
  BlocklyStorage.httpRequest_ = new XMLHttpRequest();
  BlocklyStorage.httpRequest_.name = name;
  BlocklyStorage.httpRequest_.onreadystatechange =
      BlocklyStorage.handleRequest_;
  BlocklyStorage.httpRequest_.open('POST', url);
  BlocklyStorage.httpRequest_.setRequestHeader('Content-Type',
      'application/x-www-form-urlencoded');
  BlocklyStorage.httpRequest_.send(name + '=' + encodeURIComponent(content));
};

/**
 * Callback function for AJAX call.
 * @private
 */
BlocklyStorage.handleRequest_ = function() {
  if (BlocklyStorage.httpRequest_.readyState == 4) {
    if (BlocklyStorage.httpRequest_.status != 200) {
      BlocklyStorage.alert(BlocklyStorage.HTTPREQUEST_ERROR + '\n' +
          'httpRequest_.status: ' + BlocklyStorage.httpRequest_.status);
    } else {
      var data = BlocklyStorage.httpRequest_.responseText.trim();
      if (BlocklyStorage.httpRequest_.name == 'xml') {
        window.location.hash = data;
        BlocklyStorage.alert(BlocklyStorage.LINK_ALERT.replace('%1',
            window.location.href));
      } else if (BlocklyStorage.httpRequest_.name == 'key') {
        if (!data.length) {
          BlocklyStorage.alert(BlocklyStorage.HASH_ERROR.replace('%1',
              window.location.hash));
        } else {
          BlocklyInterface.setCode(data);
        }
      }
      BlocklyStorage.monitorChanges_();
    }
    BlocklyStorage.httpRequest_ = null;
  }
};

/**
 * Start monitoring the workspace.  If a change is made that changes the XML,
 * clear the key from the URL.  Stop monitoring the workspace once such a
 * change is detected.
 * @private
```

```
 */
BlocklyStorage.monitorChanges_ = function() {
  var startCode = BlocklyInterface.getCode();
  function change() {
    if (startCode != BlocklyInterface.getCode()) {
      window.location.hash = '';
      BlocklyInterface.getWorkspace().removeChangeListener(bindData);
    }
  }
  var bindData = BlocklyInterface.getWorkspace().addChangeListener(change);
};

/**
 * Present a text message to the user.
 * Designed to be overridden if an app has custom dialogs, or a butter bar.
 * @param {string} message Text to alert.
 */
BlocklyStorage.alert = function(message) {
  window.alert(message);
};

```