Apps Script

Google-hosted Javascript

Sergii Kauk
Freelance Web Developer

"Google Apps Script provides the ability to automate a variety spreadsheet actions, such as reading and changing values in cells and ranges, changing formats and formulas, and creating custom functions. It also reaches outside of spreadsheets to allow you to send email or create calendar entries."

Jonathan Rochelle
Product Manager, Google Docs
29 May, 2009

Timeline

- Introduced in May 2009 in preview as a scripting layer for Spreadsheets
- Released to the public in August same year
- In May 2010 gets new services, standalone invocation, triggers, 1 I/O session
- One year later, in 2011, integrated in Sites, gets Docs support, format conversation, UI builder, 2 I/O sessions
- Summer 2012, mayor update with Content services, ScriptDB, publishing to Webstore and 3 talks on I/O.

What is Apps Script today?

Cloud scripting platform based on ECMAScript 5 standard with the whole development cycle inside the browser.

Let's get started!

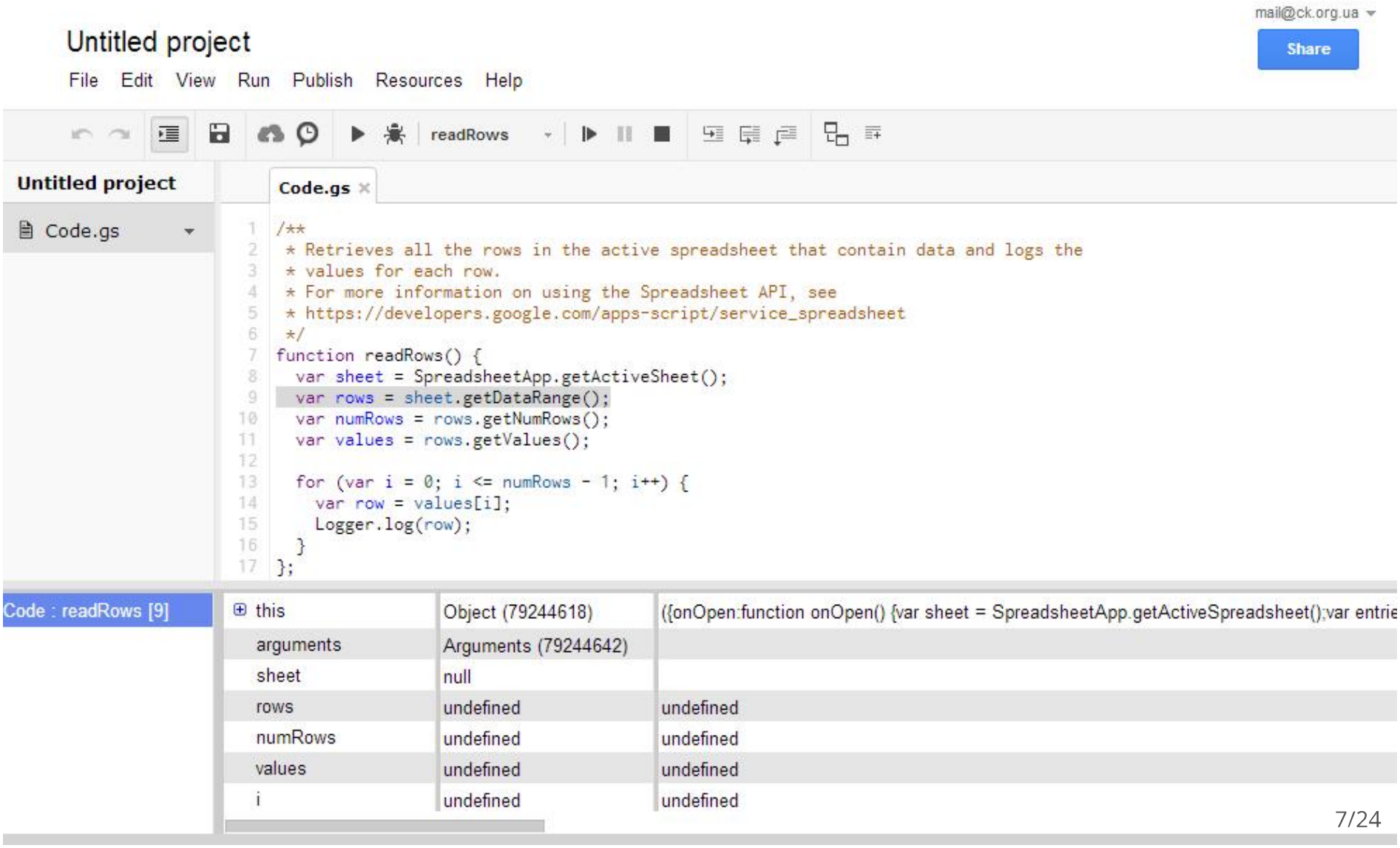
Open script.google.com

Types of scripts:

- Container-bound scripts
- Standalone scripts

```javascript
function helloWorld() {
  var message = "Hello world!";
  try { Browser.msgBox(message); }
  catch(e) { Logger.log(message); }
}
```

Editor


mail@ck.org.ua
Share
Untitled project
File Edit View Run Publish Resources Help
readRows
Untitled project
Code.gs
Code.gs
1 /**
2 * Retrieves all the rows in the active spreadsheet that contain data and logs the
3 * values for each row.
4 * For more information on using the Spreadsheet API, see
5 * https://developers.google.com/apps-script/service_spreadsheet
6 */
7 function readRows() {
8 var sheet = SpreadsheetApp.getActiveSheet();
9 var rows = sheet.getDataRange();
10 var numRows = rows.getNumRows();
11 var values = rows.getValues();
12
13 for (var i = 0; i <= numRows - 1; i++) {
14 var row = values[i];
15 Logger.log(row);
16 }
17 };
Code : readRows [9]
this Object (79244618) ({onOpen:function onOpen() {var sheet = SpreadsheetApp.getActiveSpreadsheet();var entrie
arguments Arguments (79244642)
sheet null
rows undefined undefined
numRows undefined undefined
values undefined undefined
i undefined undefined

Security model

Script is running in **Execution Context**, a security sandbox where it has access only to authorized resources.

Authorizations are revokable through Google Account settings:
accounts.google.com/IssuedAuthSubTokens

You are explicitly granting access rights to each resource used in the script.

Security model

Default permissions

- Formula-invoked function can't run code that requires authorization.
- Conteiner-bound scripts are automatically granted access to their containers.

Services

- Tool services: Base, `CacheService`, `ContentService`, `HtmlService`, `LockService`, `MailApp`, Properties, `ScriptApp`, `ScriptDb`, `UiApp`, `Utilities`
- Google Apps Services: `CalendarApp`, `Charts`, `ContactsApp`, `DocsList`, `DocumentApp`, Domain management, FinanceApp, GmailApp, GroupsApp, LanguageApp, Maps, SitesApp, SpreadsheetApp
- Google API Services: AdSense Management API, Prediction API, Tasks API, URL Shortener API, BigQuery API, Analytics API
- External services: `UrlFetch`, `Jdbc`, `SOAP`, `Xml`, `Utilities.jsonParse()`

Execution methods

- Manually invoked from editor or Spreadsheet Tool menu
- As a formula in Spreadsheets (only from Spreadsheet-bound scripts)
- Container extensions:
  - Custom menu in Spreadsheets using onLoad event
  - Assigning scripts to images in Spreadsheets
  - Assigning link target to a function in Sites
- Triggers:
  - Time-driven
  - Spreadsheet specific: onOpen, onEdit, onFormSubmit, onInstall
- Web app: `doGet()`, `doPost()`

User Interface

GWT-based

- Generated in code

APPS SCRIPT

```
function doGet() {
  var myapp = UiApp.createApplication().setTitle('Here is the title bar');
  var mybutton = myapp.createButton('Here is a button');
  var mypanel = myapp.createVerticalPanel();
  mypanel.add(mybutton);
  myapp.add(mypanel);
  return myapp;
}
```

- Or using GUI Builder

APPS SCRIPT

```
function doGet() {
  var app = UiApp.createApplication();
  app.add(app.loadComponent("MyGui"));
  return app;
}
```

User Interface

HTML Service

- Template engine

APPS SCRIPT

```
My favorite Google products!
<? var data = ['Gmail', 'Docs', 'Android'];
  for (var i = 0; i < data.length; ++i) { ?>
  <b><?= data[i] ?></b>
<? } ?>
```

- Use `google.script.run` to access functions
- Add callbacks to code execution:

HTML+JAVASCRIPT

```
<input type='button' value='Click me'
  onclick='google.script.run
    .withSuccessHandler(callbackOk).withFailureHandler(callbackFail)
    .withUserObject(this).intFu();'>
```

User Interface

Content Service

Instead of returning a UI object, the ContentService returns raw textual content of various mime-types.

APPS SCRIPT

```
function doGet(request) {
  var events = CalendarApp.getEvents(
    new Date(Number(request.parameters.start) * 1000),
    new Date(Number(request.parameters.end) * 1000));
  var result = {
    available: events.length == 0
  };
  return ContentService.createTextOutput(JSON.stringify(result))
    .setMimeType(ContentService.MimeType.JSON);
}
```

Data storage

Spreadsheets

- `SpreadsheetApp.getActiveSheet.getDataRange().getValues()` returns all data from the active spreadsheet in an array
- `setValues()` is optimized for performance, use `flush()` to manually update write cycle
- Limitations: 256 columns, 200 sheets, max. 400K cells across all sheets with only 40K with formulas (don't try that at home). Other limits on certain formulas.

Data storage

Properties

- Key/value storage, string data only
- User and Script properties with according permissions
- There is a maximum size limit of 500kB total for all properties in a script. Each property value has a maximum size limit of 9kB.

Data storage

ScriptDb

ScriptDb is a JavaScript Object database.

- Multiple key/value pairs per record
- Database instance is tied to a script
- Transactions can be emulated using Lock service

APPS SCRIPT

```
var db = ScriptDb.getMyDb();
var ob = {type: "employee",
          employee_id: 1,
          name: {first: "Otto", last: "Bauer"},
          address: {street: "Ottakringerstrasse 56",
                    city: "Vienna", zip: "1160"},
          department_id: 52};
var stored = db.save(ob);
```

Data storage

JDBC Service

- MySQL

APPS SCRIPT
```
var sqlGoogleConnection = Jdbc.getCloudSqlConnection
("jdbc:google:rdbms://instance_name/database_name", "user", "password");
```

- Cloud SQL

APPS SCRIPT
```
var sqlMyConnection = Jdbc.getConnection("jdbc:mysql://:/",
"user", "password");
```

More info: developers.google.com/cloud-sql

Publishing

- Script Library
- Web app with simplified publishing to Chrome Webstore
  - Versions represent a snapshot of the script that can be published and simplify new releases
- Libraries

Conslusion

- Google Apps Script matured to the point where it can compete with many common web development tools and offer much faster and easier development cycle.

New lamp is even better

https://www.youtube.com/watch?v=I07xDdFMdgw

Resources

- Documentation: developers.google.com/apps-script
- Release notes: developers.google.com/apps-script/release_notes
- Issue tracker: code.google.com/p/google-apps-script-issues/issues
- Eric Koleda +1

<Thank You!>

g+ plus.google.com/106401792982259937659