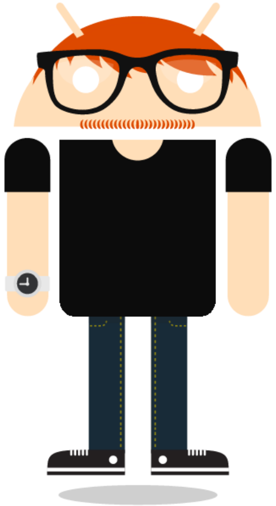
Google™ 11 IO

Using The Google Docs APIs To Store All Your Data in the Cloud

Vic Fryzel - @vicfryzel
May 11, 2011

#io2011 #googleapps



Cloud storage is a revolution in the abundance and availability of user data.

Developers must be able to assist users, but need rich, reliable APIs.

Users need to access their data in more ways, from more devices and more locations.

A cloud storage API must deliver more than just storage

- **Sharing** with international friends and co-workers is a key user need

- **Data** must be consumable in useful forms

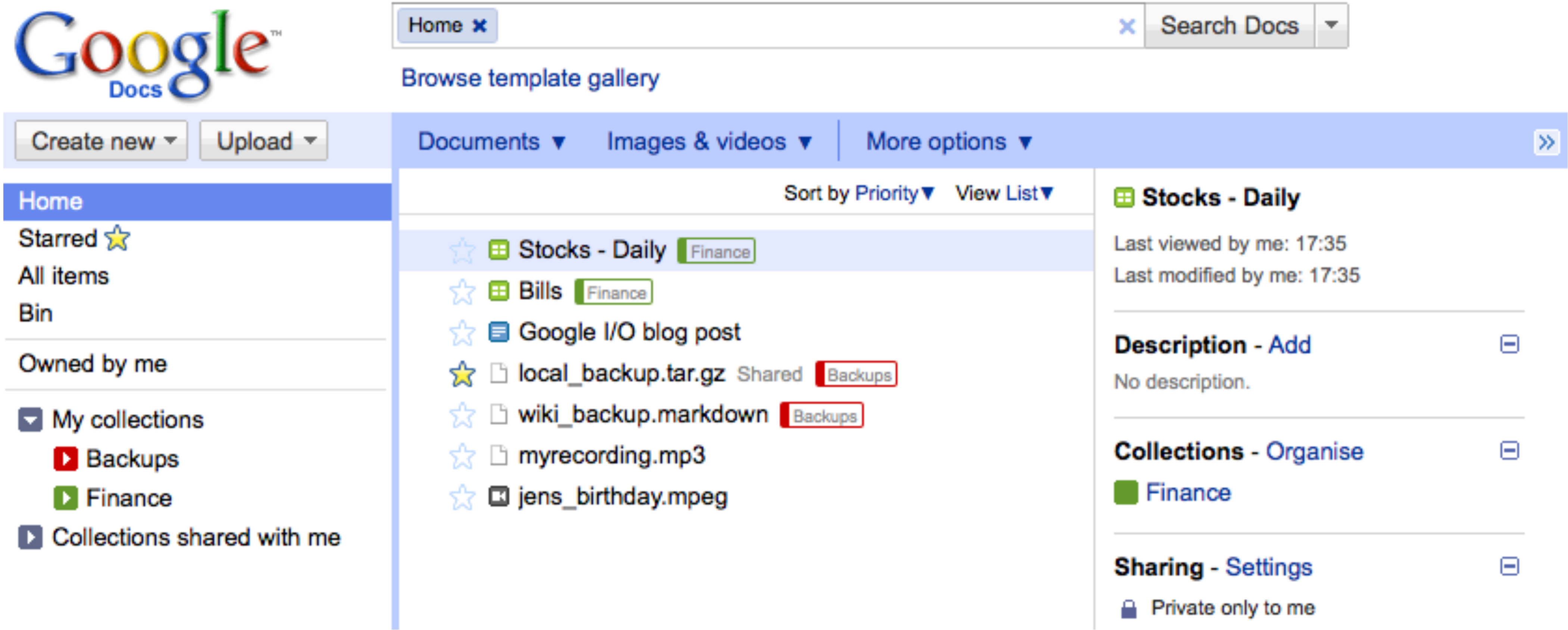
- **Web UIs** are needed in many cases as an alternative way to access data
- **Cloud services** should automate a lot of tasks so that developers don't have to re-implement the wheel

The Google Documents List API is one example of such a storage API.

But the API stores more than just documents.

Announcing **uploads of arbitrary file types** for all Google users via the API.

Using the API for more than just Docs



Stocks - Daily		Finance
Bills		Finance
Google I/O blog post		
local_backup.tar.gz	Shared	Backups
wiki_backup.markdown		Backups
myrecording.mp3		
jens_birthday.mpeg		

Stocks - Daily

Last viewed by me: 17:35

Last modified by me: 17:35

Description - Add

No description.

Collections - Organise

🟩 Finance

Sharing - Settings

Private only to me

Different than other storage APIs

- Storage is per user; users control storage quota
- Data is inherently structured
- All entries have the same metadata
- Documents currently use zero quota

How do you use this in a *real* application?

Examples of *real* applications

Spanning Backup
for Google Apps

spanningbackup.com

syncplicity

syncplicity.com

mĕmeo

memeo.com

Google Docs
app for Android

Android Marketplace

Quick review of basic API terms

- **Document** is anything editable in Google Docs
- **File** is anything not editable in Google Docs
- **Collection** organizes other resources
- **Resource** is a document, file or collection

What technologies should our app use?

OAuth to get a user's *permission* to access their documents list.

App Engine to host the application.

JavaScript and HTML5 to give a usable interface.

Documents List API to manage most data. Use the Spreadsheets API for data with a custom schema.

What does the application do?

Walking through the source code

- Create an **App Engine** app
- Implement **3-Legged OAuth**
- Implement a **JavaScript and HTML5** UI
- Use the **Documents List API** to manage everything
- **~375 total lines of Python & JavaScript**

Follow along at home!

http://goo.gl/NAkZM

Best practices

- Use resumable upload for everything
- Fetch as many resources as possible in a single request
- Cache each resource individually
- Perform collection tree generation server side
- Cache collection trees and their contents
- Use the app:edited date to determine when a resource was last edited by a user

Best practices continued

- Keys set initially on "share with key" ACLs will be automatically overridden
- Batch ACL requests together into a single request
- To share a large number of items at once, add them all to a collection, then share the collection
- When needing to back-up an employee's docs list, use admin access to impersonate them with the API
- Use OAuth whenever possible, especially instead of ClientLogin

Consider the Documents List API for cloud storage.

More resources, questions, answers

- **Documentation**
  - goo.gl/ATu6x
- **Forum**
  - goo.gl/ozJiZ
- **Code**
  - goo.gl/NAkZM
- **Thanks to**
  - Rob Wyrick
  - Russ Jorgensen
  - Ivan Lee
  - Google Docs team
- **#io2011 #googleapps**

Google™ 11 I O