SeedMe

Your results from disk to device

Continuous and Ubiquitous Sharing via SeedMe.org

Amit Chourasia, Sr. Visualization Scientist
San Diego Supercomputer Center, UCSD
Tutorial Requirements

1. Create account on http://www.seedme.org
   My Account > Create new account

2. Python version 2.6 or 2.7 (preferably) or 3.x
   (Python knowledge not required)

3. Curl executable (Optional)
Pitfalls in sharing derived content

- Download Upload Download (Round Trip + 1)
- Video Encoding Complexity
- Time Delay
- Web Setup Process Duplication
- Scalability
SeedMe: How it Works

1. **SignUp**
   - **Web Browser**

2. **Create Collection**
   - **Command Line or REST (Process Automation)**

3. **Set Privacy**

4. **Set Metadata**
   - **Update as needed**

5. **Upload Content**
   - **Phone or Tablet or Computer**

6. **View on Web**
SeedMe Collection

- **Ticker**
  - (Text)
  - Ephemeral

- **Files**
  - (non image)

- **Plots**
  - (Image)

- **Sequences**
  - (Image set)

- **Videos**
  - (video)

**Privacy + Collaborators**
- (Public, Group, Private) + (emails)

**Meta Data**
- (Title, Description, Tags, Key Value Pairs, Credits, License)
Acknowledgements

SeedMe.org Team Members
Mona Wong-Barnum, Michael Norman

Andrew Ferbert, Kristen Levy, Michael Dwyer, & Doug Weimer, SDSC: Hardware/software setup
Mahidhar Tatineni, SDSC, UCSD: in-situ tests
Apple Inc.: Provided test hardware/software on loan during project conceptualization phase

National Science Foundation
This material is based upon work supported by the National Science Foundation under Grant No. OCI-1235505
SeedMe.org
Quick start using Command Line
# SIGN-UP at SeedMe.org
https://www.seedme.org/user/register

1. Log in to SeedMe > My Account
2. Download Authorization file: *seedme.txt*
   Move *seedme.txt* file to your home folder

<table>
<thead>
<tr>
<th>Operating System</th>
<th>Home Folder Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>Windows 2000, XP and 2003</td>
<td>C:\Documents and Settings&lt;username&gt;</td>
</tr>
<tr>
<td>Windows Vista, 7 and 8</td>
<td>C:\Users&lt;username&gt;</td>
</tr>
<tr>
<td>Mac OS X</td>
<td>/Users/&lt;username&gt;</td>
</tr>
<tr>
<td>Unix-Based</td>
<td>Varies, check in terminal as follows</td>
</tr>
<tr>
<td></td>
<td>% echo $HOME</td>
</tr>
<tr>
<td></td>
<td>OR</td>
</tr>
<tr>
<td></td>
<td>% cd ~ ; pwd</td>
</tr>
</tbody>
</table>
# SeedMe Terminology

Collections are a container of any of the following elements

- **METADATA**: Title, Description, Key Value Pairs, Credits, License
- **TICKER**: Short text string (128 chars)
- **FILES**: Non image files
- **PLOTS**: Unrelated images
- **SEQUENCES**: Set of related images

Each collection is automatically assigned a numeric identifier as **collection_id**
# Download SeedMe module
a) Documentation > E. Command Line Usage > #2
OR
b) http://www.seedme.org/sites/seedme.org/files/seedme_module/seedme.zip

# Requirements
Python version 2.6 or 2.7 (preferably) or 3.x
(Python knowledge not required)
# CREATE A COLLECTION

```python
python seedme.py -title "SeedMe CLI Quick Start"
```

-------------
Sample Output (d0)
-------------
Uploading chunk 1 of 1
Attempting to create a new collection
Success: Collection created at collection id 29643
{"collection_id":"29643","status":"success"}
-------------

# Note: "collection_id" : "29643"
# This will be needed for updating collection
# CHANGE COLLECTION PRIVACY

# Update this new collection to be group accessible with two people
# Note: By default collections are private

```python
python seedme.py -update 29643 \
    -privacy group \
    -email amit@sdsc.edu \
    -email doe@sdsc.edu
```

-------------
Sample Output (dl)
-------------
Uploading chunk 1 of 1
Success: Collection updated at collection id 29643
{"collection_id":"29643","status":"success"}
-------------
# ADD KEY VALUE PAIRS TO THIS COLLECTION

```python
python seedme.py -update 29643 \ 
  -keyvalue "system: gordon" \ 
  -keyvalue "exe:/scratch/enzo" \ 
  -keyvalue "indata:/scratch/input" \ 
  -keyvalue "outdata:/scratch/output" \ 
  -keyvalue "pressure:10pa" \ 
  -keyvalue "temperature:1000K"
```

Sample Output (d2)

---------

Uploading chunk 1 of 1
Success: Collection updated at collection id 29643
{"collection_id":"29643","status":"success"}
---------
# ADD TICKERS TO TRACK SIMULATION PROGRESS
(Simulate progress with 1-second delay)

```bash
#!/usr/bin/bash
for i in {1..10};
  do myticker="Step $i";
  echo $myticks;
  python seedme.py -update 29643 -ticker "$myticks";
  sleep 1;
done
```

# Note: Enclose the ticker text with quotes to deal with space, etc

-------------
Sample Output d3
-------------
Step 1
Uploading chunk 1 of 1
Success: Collection updated at collection id 29643
{"collection_id":"29643","status":"success"}
...
[snipped]
...
Step 10
Uploading chunk 1 of 1
Success: Collection updated at collection id 29643
{"collection_id":"29643","status":"success"}
-------------
# VIEW COLLECTION IN WEB BROWSER

# Computer
# Tablet
# Phone

# https://www.seedme.org/collection/29643
# ADD IMAGE SEQUENCE, CREATE A VIDEO FROM IT

```python
python seedme.py -update 29643 \
    -sequence_path "sample/sequences/air/*" \
    -sequence_title "My sequence title" \
    -sequence_description "Desc. of sequence" \
    -sequence_frame_rate "5" \
    -sequence_encode
```

-------------
Sample Output d4
-------------
Uploading chunk 1 of 1
Success: Collection updated at collection id 29643
{"collection_id":"29643","status":"success"}

-------------

# Note: Possible to append to a sequence periodically
Just pass collection id and sequence title
Trigger video encoding after last append if needed
# VIEW COLLECTION IN WEB BROWSER

# Computer
# Tablet
# Phone

# https://www.seedme.org/collection/29643
# Notify Collaborators

`python seedme.py -update 29643 -notify`

---------
Sample Output d5
---------
Uploading chunk 1 of 1
{"collection_id":"6994", "status":"partial",
"notify_message":"The following collaborator has been notified: amit.
The following inactive collaborator has NOT been notified: doe@sdsc.edu."}
WARNING: Partial: Incomplete Collection update at collection id 6994

# Note: Notification is not automatic.
You decide when collection is ready for sharing

# Note: Partial success
Currently, only active members notified at present
# OTHER OPTIONS

# Perform dry run to validate input
Add Option `-dry` or `--dry_run`

# Show curl commands for a given input
Add Option `-s` or `--show_curl_commands`
# CURRENT LIMITATIONS
- Cannot retrieve collections via web services
  (need use case to consider implementation)
- Cannot delete collections via web services

# ENHANCEMENTS IN WORK
- WYSIWYG creation, update and deletion of collection
- Overwrite existing files
- Expiry Date – Automatic purge after a specified period
# WHAT ABOUT AUTHENTICATION?

Web Browser uses **username** and **password** authentication

Web Services uses **username** and **api_key** authorization

Preferred method is to store authentication info to a text file in JSON format at ~/.seedme or ~/.seedme

Note: Secure this file by making it only user readable on linux % chmod 600 ~/.seedme.txt

```
cat ~/.seedme.txt
{
  "username" : "YourUserName",
  "api_key" : "YourApikey"
}
```

Alternatively, authentication credentials could be passed as arguments
```
python seedme.py -username "YourUserName" \
  -api_key "YourApikey"
```

All communications are sent via https
# WHAT ABOUT NON-CLI USAGE?

Python module currently available
(seedme.py is a wrapper to the seedme module)

Only two functions to interact with SeedMe.org

```python
createCollection(view_permission='', viewer_emails='', title='',
description='', credits='', license='', tags=[], tickers=[],
key_values={}, files={}, plots={}, sequences={}, videos={})

updateCollection(collection_id, view_permission='',
viewer_emails='', title='', description='', credits='',
license='', tags=[], tickers=[], key_values={}, files={}, plots={},
sequences={}, videos={})
```

SeedMe offers simplified REST service that may be used from other tools with relative ease
# Integration with FORTRAN

# Fortran 90
CALL SYSTEM('python /path/seedme.py -dry -t "test title" &')
Note: Make sure to append the system command with "&" for non-blocking call.

Fortran 2008
CALL execute_command_line('python /path/seedme.py -dry -t "test title"', WAIT=False)
Note: The above is a non-blocking call when WAIT=False
# Integration with C

fork('python /path/seedme.py -dry -t "test title"')

exec('python /path/seedme.py -dry -t "test title"')

system('python /path/seedme.py -dry -t "test title" &')

Note: Make sure to append the system command with "&" for non-blocking call.
# WHAT CONTENT CAN BE UPLOADED?

- Text: Tickers, title, description, etc
- Files: pdf, ipynb
- Plots: png, jpg
- Sequence: png, jpg
- Video: mp4, mov, m4v

# RESULTS ONLY, NO DATA PLEASE
Q & A

Expecting SeedMe in production by XSEDE14 (July 2014)