New Features in SAGE2 v3

November 2017

SAGE2Cloud 3
Introduction 3
Prerequisites 3
Installation 4
Edit configuration file 4
Setup the SSL certificate 5
   If you don't have a certificate, then you can generate a self-signed one (Linux only instructions): 5
pm2 cheat-sheet 5
Youtube video 6
Credits 6

SAGE2 Unity3D Applications 7
Introduction 7
Unity Build Configuration 7
Preparing to install on SAGE2 8
Interaction 9
   SAGEPointer 9
   Interacting Directly on Display Client 10

Jupyterlab Integration 11
Extension 11
   Disclaimer: A SAGE2 server must be running version 2.0.327 or newer to use this JupyterLab functionality. 11
Package Installation 13
  Prerequisites 13
  Installation 13
  Development 13
Docker Installation 14
  To install/update 14
  To run 14
  Future Plans 14
  Issues and Contributing 14
  Youtube video 14

Performance monitoring for SAGE2 15
  Performance page 15
  Metrics and Graphs 15

Voice Commands 19
  Common Commands 19
  Wall Commands 20
  Application Commands 21
  UI client Commands 22
  Youtube video 22

Applications Updates 23
  Movie Sync Remote Control 23
  ChemViewer Updates 23
  Notepad Update 24

Application Data Sharing 24
  Youtube Video 24

Remote Pointers 25
  Youtube Video 25

User Model 26
SAGE2Cloud

SAGE2Cloud is a server for creating SAGE2 docker containers. It simplifies SAGE2 deployment:

- One click SAGE2 server installation and configuration
- Monthly SAGE2 version selector
- One-click updater
- Support for multiple users and machines

It provides a scalable environment for deploying and managing multiple SAGE2 servers.

Introduction

SAGE2Cloud is a management system for SAGE2 servers. With SAGE2Cloud you can:

- Create, deploy and configure SAGE2 local and remote servers
- Edit and customize the configuration
- Start/Stop the SAGE2 servers

Prerequisites

- docker
- Mongodb
- Install pm2 to manage the Node.js instance:
  #!bash
  npm install pm2@latest -g
Installation

Add the user that will be running docker into the docker group:

```
#!bash
sudo vi /etc/group
```

Run 'docker ps' to make sure you can connect to docker.

Make sure docker daemon is running: on a machine with systemctl, use:

```
systemctl status docker
```

Clone the repository:

```
#!bash
git clone https://bitbucket.org/sage2/sage2cloud
cd sage2cloud
mkdir certs
npm run in
```

For SAGE2Cloud development instances, you can disable mongodb authentication by editing '/etc/mongodb.conf' and commenting out the lines starting with 'security' and 'authorization'. If you have set up mongodb with authentication, then make sure you provide the right credentials in config.js (see below).

Make sure that 'mongod' is running: on a machine with systemctl, use:

```
systemctl status mongod
```

Edit configuration file

```
#!bash
cp config.js.example config.js
vi config.js
```

- **hostname**: The domain where sage2cloud is running. For example, www.sage2cloud.com. Make sure you have the DNS set up correctly.
- **http**: The http port that sage2cloud will be running.
- **https**: The https port that sage2cloud will be running.
- **mongoHost**: URL where mongodb is running. If your mongodb is running on the same machine as sage2cloud, leave it as localhost. If you are not sure, leave it as is.
- **mongoPort**: Port where mongodb is running. Default is 27017. If you are unsure, leave it as is.
- **mongoName**: Name of the db in mongodb. For most instances, sage2Cloud should work.
- `mongoUser`: If `mongodb` authentication is enabled, here you have to enter the username. If not, leave it as is.
- `mongoPass`: If `mongodb` authentication is enabled, here you have to enter the password. If not, leave it as is.
- `useMongoAuthentication`: true if authentication is enabled (don't forget to set username and password above), false for no authentication.
- `certs`: Ca, cert and key certificates. These files should be in the certs directory. See below for instructions on creating certificates if you don't have any.

### Setup the SSL certificate

Copy `certs` in the certs directory. NOTE: Certificates need to follow the following naming convention:

```bash
#. <domain>-ca.crt
#. <domain>.crt
#. <domain>.key
```

If you don't have a certificate, then you can generate a self-signed one (Linux only instructions):

```bash
# !bash
wget https://bitbucket.org/sage2/sage2/raw/8f4b44f1e24068a24cba8f4462838bf6adbdf39/keys/init_webserver.sh
chmod 755 init_webserver.sh
./init_webserver.sh <ip or hostname>
```

For example, if you are setting up `sage2cloud` for `google.com`, then your keys should be named as follows:

```bash
# !bash
_.google.com-ca.crt
_.google.com.crt
_.google.com.key
```

### pm2 cheat-sheet

Start app

```bash
# !bash
pm2 start app.js
```

Stop app
```bash
pm2 stop app.js
```

Add it to the startup
```bash
sudo pm2 startup systemd
```

Run as sage user
```bash
sudo pm2 startup -u sage
```

Reload pm2 (start/stop the pm2 daemon with all the apps that's controlling)
```bash
sudo pm2 update
```

**Youtube video**

[https://youtu.be/q6AdaeoFDm8](https://youtu.be/q6AdaeoFDm8)

**Credits**

Account management backbone forked from [Node-Login](https://github.com/stephenbraitsch/node-login) created by Stephen Braitsch.
SAGE2 Unity3D Applications

Introduction

Unity WebGL applications can be displayed in SAGE2 similar to how native SAGE2 applications - by dragging and dropping a zipped application folder on the SAGE2 web UI. As Unity applications are displayed as locally hosted WebViews, the display client must be using Electron. WebGL builds from Unity 5.4.2 up to Unity 2017.1.1 have been tested.

Unity applications will be listed under the Media Browser along with the other SAGE2 applications.

This an example SAGE2 Unity app is available on the SAGE2 App Repository (http://apps.sagecommons.org) as 'Unity Interaction Test' under 'Examples for Developers'

The Unity project source code is also available at: https://bitbucket.org/sage2/sage2_unity

Unity Build Configuration

Before a Unity WebGL application is built for use in SAGE2, under Build Settings/Player Settings/Settings for WebGL/Resolution and Presentation, set the WebGL Template to
'Minimal.' This will format the application so it can be displayed as a full screen, window-scalable application.

Preparing to install on SAGE2

Native SAGE2 applications require an instructions.json file which specifies the size of the window, custom icons, description, author, and more. For Unity applications, having an instructions.json file is recommended, but not required. If SAGE2 detects a Unity WebGL application without an instructions.json, it will be auto generated.
Example instructions.json

```json
{
    "main_script" : "UnityLoader.js",
    "icon" : "unity.png",
    "width" : 960,
    "height" : 600,
    "resize" : "free",
    "animation" : "true",
    "dependencies" : [],
    "load" : {
    },
    "title" : "SAGE2-Unity-Application",
    "version" : "1.0.0",
    "description" : "Unity WebGL build loaded as a Webview",
    "keywords" : ["sage2", "unity3d", "webview"],
    "author" : "Name <email@mail.com>",
    "license" : "SAGE2-Software-License"
}
```

Once the Unity WebGL build is generated zip the folder so that the zip is in the following structure

- [Application folder (Same name as zip file)]
  - Build folder
    - [Unity data files]
    - UnityLoader.js
  - index.html

**Interaction**

Some keys such as Escape, Middle Mouse Button, and Shift-QuestionMark are used by SAGE2 and may conflict with interaction.

**SAGEPointer**
### Interacting Directly on Display Client

<table>
<thead>
<tr>
<th>Interaction (Mouse)</th>
<th>Support Status</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input.mousePosition</td>
<td>Supported</td>
<td>Mouse0 (Left), Mouse1 (Right), Mouse2 (Middle) all working. Note Mouse2 is also used to switch interaction mode.</td>
</tr>
<tr>
<td>Input.keycode (Mouse)</td>
<td>Supported</td>
<td>Mouse0 (Left), Mouse1 (Right), Mouse2 (Middle) all working. Note Mouse2 is also used to switch interaction mode.</td>
</tr>
<tr>
<td>Input.keycode (Key)</td>
<td>Partial Support</td>
<td>No lowercase letters*. Some non-alpha numeric keys missing: F-keys, Home, End, Insert, Delete, Minus, Backquote, LeftBracket, Semicolon, Comma, Period, Quote</td>
</tr>
<tr>
<td>Input.scrollWheelDelta</td>
<td>Supported</td>
<td></td>
</tr>
<tr>
<td>InputManager</td>
<td>Partial Support</td>
<td>&quot;Horizontal&quot; and &quot;Vertical&quot; partially working*, &quot;Fire&quot; 1, 2, 3, &quot;Jump&quot;, and &quot;Submit&quot; fully working.</td>
</tr>
<tr>
<td>Canvas Interaction (Buttons, Slider, Toggle)</td>
<td>Supported</td>
<td></td>
</tr>
<tr>
<td>Canvas Text Field</td>
<td>Partial Support</td>
<td>Keys not supported by Input.keycode work, except no backspace or lower case letters*.</td>
</tr>
<tr>
<td>Gamepads</td>
<td>Not Supported</td>
<td>Future feature</td>
</tr>
</tbody>
</table>

### Interacting Directly on Display Client

<table>
<thead>
<tr>
<th>Interaction</th>
<th>Support Status</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input.mousePosition</td>
<td>Supported</td>
<td></td>
</tr>
<tr>
<td>Input.keycode (Mouse)</td>
<td>Supported</td>
<td>Mouse0 (Left), Mouse1 (Right), Mouse2 (Middle) all working. Note Mouse2 is also used to switch interaction mode.</td>
</tr>
<tr>
<td>Input.keycode (Key)</td>
<td>Partial Support</td>
<td>No lowercase letters*.</td>
</tr>
<tr>
<td>Input.scrollWheelDelta</td>
<td>Supported</td>
<td></td>
</tr>
<tr>
<td>InputManager</td>
<td>Supported</td>
<td>No lowercase letters*, backspace supported</td>
</tr>
<tr>
<td>Canvas Interaction (Buttons, Slider, Toggle)</td>
<td>Supported</td>
<td></td>
</tr>
<tr>
<td>Canvas Text Field</td>
<td>Supported</td>
<td>No lowercase letters*, backspace supported</td>
</tr>
<tr>
<td>Gamepads</td>
<td>Supported</td>
<td>via native Gamepad API for HTML5</td>
</tr>
</tbody>
</table>

- All letter keys are sent to Unity applications as uppercase to allow InputManager Vertical/Horizontal axis to function though the SAGEPointer
Jupyterlab Integration

A JupyterLab extension to integrate SAGE2 into the JupyterLab scientific workflow.

Three researchers using JupyterLab and SAGE2 together with the jupyterlab_sage2 extension.

Extension

The goal of this extension is to use SAGE2 with JupyterLab to integrate existing data science workflows into the collaborative software.

Disclaimer: A SAGE2 server must be running version 2.0.327 or newer to use this JupyterLab functionality.

This extension to JupyterLab allows a user to simultaneously connect to multiple SAGE2 servers and share JupyterLab content with SAGE2, including:

- Notebooks
- Notebook Cells (Images)

From the JupyterLab launcher, open the SAGE2 widget under "Other." Once the SAGE2 Widget is opened, a server connection can be created by clicking the green '+' button and entering the server Address. A Server Name can be specified in order to more easily refer to a server later.
The plugin displays current server connections -- these connections are persistent whether or not the widget is open. The gold star indicates a 'favorite' SAGE2 server, or a server which you will be using more frequently.

When a Notebook is open, the Notebook or a selected cell output may be sent to SAGE2 through the main menu. Sending a Cell/Notebook to 'Favorite' sends the content to the
Favorited server, while sending content to '...' opens a dialog in which a user may choose one of the SAGE2 connections to send to.

Notebooks are sent to SAGE2 and rendered using nbviewer. Notebook cells are rendered as images and automatically updated when a cell is re-run.

**Package Installation**

**Prerequisites**
- JupyterLab

**Installation**

```bash
jupyter labextension install jupyterlab_sage2
```

**Development**

For a development install (requires npm version 4 or later), do the following in the repository directory:

```bash
npm install
ejupyter labextension link
```

To rebuild the package and the JupyterLab app:

```bash
npm run build
ejupyter lab build
```
Docker Installation

To try JupyterLab with SAGE2, use the sage2/jupyterlab-datascience-notebook Docker image. This image is built FROM the jupyter/datascience-notebook and includes Python 3, R, and Julia as well as a variety of data science packages.

To install/update

docker pull sage2/jupyterlab-datascience-notebook

To run

docker run -it --rm -p 8888:8888 sage2/jupyterlab-datascience-notebook start.sh jupyter lab

The -p 8888:8888 maps the external port to the docker port: -p external:internal. If you would like to select a different port to access Jupyterlab use -p yourport:8888.

When starting the docker image, the console will give a JupyterLab url with an access token.

To access JupyterLab, navigate your webpage to


For more information on the jupyter/datascience-notebook Docker image, visit:
https://github.com/jupyter/docker-stacks/tree/master/datascience-notebook

Future Plans

In the future, we plan to support more content types and methods of sending data from JupyterLab to SAGE2, as well as implement a file browser for JupyterLab which allows access to SAGE2 files in the JupyterLab workspace.

Issues and Contributing

- Please direct any issues or bug reports to the repository's Issues
  - https://github.com/AndrewTBurks/jupyterlab_sage2/issues
- If you would like to contribute, submit a Pull Request
  - https://github.com/AndrewTBurks/jupyterlab_sage2/pulls

Youtube video

https://youtu.be/l2pRLhw6GSE
Performance monitoring for SAGE2

The new performance monitoring utility provides a comprehensive view of the health of SAGE2 system enabling the users to narrow down the cause of potential issues like slow responses to interactions or a video becoming “choppy”.

Performance page

The performance monitoring page is listed as “Performance” under the “Advanced” menu at the right top corner of the SAGE2 UI page as shown in Fig1.

![Fig1: Accessing Performance page through SAGE2 UI](image)

Metrics and Graphs

The page has two sections, one for the server and one for the clients. Fig2 gives an overview of what the page looks like. On the server side of things, the hardware details of the machine on which the SAGE2 server is running is first listed out. After that, three different metrics namely Load, Memory, and Network traffic for both the SAGE2 server and the overall host machine are
visualized as charts. These charts show the trends of each of the metrics for the past 5 minutes, along with giving out their current values. Fig3, Fig4, and Fig5 each shows these charts.

Fig2: Performance monitoring page

Fig3: Load of SAGE2 server and overall system load
Under the client section, the hardware details of all the display clients that are connected are given in a list. Then a small-multiples view of the display clients is provided, where each small-multiple shows the current values of load and memory for the corresponding display client as a horizontal histogram. Moreover, these small-multiples are interactive. Users can select one or more of these small multiples as shown in Fig. 6 to bring up metric charts similar to server metric charts. These charts show multiple lines, where each line corresponds to one of the selected display clients. The small-multiple of a selected display client is highlighted with a colored border and the same color is used to display the corresponding line in the charts.
Fig6: Multiple display clients selected to show their metric charts
# Voice Commands

## Common Commands

<table>
<thead>
<tr>
<th>Action</th>
<th>Phrase to activate</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Tile all open applications</strong></td>
<td>Tile the wall</td>
</tr>
<tr>
<td><strong>Close all open applications</strong></td>
<td>Close everything</td>
</tr>
<tr>
<td><strong>Restore closed applications</strong></td>
<td>Restore everything&lt;br&gt;Restores applications closed by the previous command</td>
</tr>
<tr>
<td><strong>Launch an application</strong></td>
<td>Open [phrase]&lt;br&gt;Opens the application with the most word matches in [phrase] compared to title, description, or keywords.</td>
</tr>
<tr>
<td><strong>Make a note</strong></td>
<td>Make a note [phrase]&lt;br&gt;Will create a note with the [phrase] as the note contents</td>
</tr>
<tr>
<td><strong>Save session</strong></td>
<td>Save session as [phrase]&lt;br&gt;Saves all open applications as session with name [phrase]</td>
</tr>
<tr>
<td><strong>Load session</strong></td>
<td>Load session [phrase]&lt;br&gt;Loads the session where the name has the most word matches in [phrase]</td>
</tr>
<tr>
<td><strong>Search on the web</strong></td>
<td>Web search for [phrase]&lt;br&gt;Starts a web search using [phrase]</td>
</tr>
<tr>
<td><strong>Image search on the web</strong></td>
<td>Web image search for [phrase]&lt;br&gt;Starts a web image search using [phrase]</td>
</tr>
<tr>
<td><strong>Share to remote site</strong></td>
<td>Share with [phrase]&lt;br&gt;Shares the application under the pointer to the remote site&lt;br&gt;name with the most words matches in [phrase]</td>
</tr>
<tr>
<td>Action</td>
<td>Invocation Words Required</td>
</tr>
<tr>
<td>---</td>
<td>---</td>
</tr>
</tbody>
</table>
| **Tile all open applications** | Clean wall  
Clean this up  
Cleanup  
Organize  
Tile content  
Tile everything  
Tile wall  
Tile windows |
| **Close all open applications** | Clear everything  
Close everything  
Get rid everything  
Toss everything  
Toss it all |
| **Restore closed applications** | Restore view  
Restore everything  
Bring back everything  
This will restore applications closed by the previous command. |
| **Launch an application**  
Open file from media browser | Launch [phrase]  
Load application [phrase]  
Open [phrase]  
Start [phrase]  
**Based on the [phrase], the application with the most matches within the tile, description, or keywords (described in instructions.json) will be launched.**  
**If any phrase contains the word: pdf, image, picture, video, or movie. Instead of an application, it will search for a name match within the corresponding folder of the media browser.** |
| **Make a note** | Make a note [phrase]  
Make a reminder [phrase]  
Write down [phrase]  
**Creates a note with the [phrase] as the contents.** |
|  | Save applications as [phrase] |
| Save session with a name | Save content as [phrase]  
| | Save session as [phrase]  
| | Save state as [phrase]  
| | Save wall as [phrase]  
| | Save applications name [phrase]  
| | Save content name [phrase]  
| | Save session name [phrase]  
| | Save state name [phrase]  
| | Save wall name [phrase]  
| | Creates a saved session with [phrase] as the name visible within the media browser. |
| Restore all applications that were part of a saved session | Restore session [phrase]  
| | Load session [phrase]  
| | Bring back [phrase]  
| | Restores a session with the most word matches within the given [phrase]. |
| Open a Webviewer application and search using Google | Web search [phrase]  
| | Google search [phrase]  
| | Will first open a new Webview application, then perform a Google search with the given [phrase]. If “image” is said after web or google, image search will be done instead of text search  
| | Ex: Google image search [phrase]. |
| Share an application to a remote site | Share with [phrase]  
| | Send to [phrase]  
| | Will share the application directly under the pointer to the remote site who’s name has the most word matches within [phrase]. |

For wall commands, each of the words of an invocation must be present. For commands with phrases, the phrase must follow the preceding word, otherwise the phrase will be incorrectly detected. Aside from the phrase and the word before it, the invocation words can be in any order.

**Application Commands**

Applications can be issues commands from their context menu. The menu entry activated will be the one with the most word matches. When multiple applications are
open, the application directly under the pointer will be checked first. If no matches are found, the rest of the applications will be checked starting from the oldest.

For entries that take user input, the word before the input area is used to identify the start of input. Anything after that word will be sent as input.

UI client Commands

The following commands can be activated from within the client UI. They do not reach the server for evaluation.

<table>
<thead>
<tr>
<th>Action</th>
<th>Invocation Words Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tell the time</td>
<td>What time is it</td>
</tr>
<tr>
<td>Tell the data</td>
<td>What is today's date</td>
</tr>
<tr>
<td>Open the help page</td>
<td>Help</td>
</tr>
<tr>
<td></td>
<td>Will open in a new tab the quick reference for voice commands. This may be prevented by pop-up blockers.</td>
</tr>
</tbody>
</table>

Youtube video

https://youtu.be/9atA_h_t5pM
Applications Updates

Movie Sync Remote Control

- This application is able to control multiple video players at once.
- Detects when over a video player and can add that player under its controls from the context menu or within the application.
- The longest connected player becomes master, which the other player will synchronize with.

ChemViewer Updates

- Expanded to show information from http://www.ebi.ac.uk/chebi/
- Updated by Jared McLean, at University of Hawaii Hilo
Notepad Update

- Now supports text scrolling
- Updated by Jirayu Roungsuriyaviboon of Mahidol University, Thailand

Application Data Sharing

- Data sharing was added to SAGE2 application API to allow multiple applications to work as one large application
  - Also enable applications to be designed like components that other developers can utilize
  - Applications are able to launch other applications
  - This allows not only data passing, but window manipulation

- Applications that launch others, are able given each others’ ID
  - The IDs are pass in variables to identify the parent (launcher) and child (launched application)

- Applications can create named variables on the server
  - Those variables can be requested or subscribed to
  - Any application can create, request, subscribe, or modify variable

Youtube Video

https://youtu.be/uU5HZtmvS10
Remote Pointers

- Improves collaboration and identifying features in an application
  - Applications shared with a remote site shows pointers over an application from participants of both sites
  - Pointers of the remote site are suffixed with "@hostname"
- Utilizes application synchronization
- Pointers are associated with the application
  - Relative position is preserved even if scale is mismatched between sites
  - Can move the application on your site and relative position will still be preserved
  - Application must be visible to see remote pointer
  - Hides pointer when in window management mode

Youtube Video

https://youtu.be/6E7pdyuhwhA
User Model

- Distinguish between people using SAGE2
  - Access and authorization
  - Log activity—who did what
  - Scenarios and roles, e.g.
    - Group meeting with a leader
    - Teachers / students
    - …etc.
Open the User Console via the Advanced menu

Example scenario: only the teacher (admin role) has permission to use the wall

Example scenario (cont’d): teacher allows other people to add files to the wall

See who’s connected
Manage roles and permissions

Actions by unauthorized users are disabled

Adding files is now enabled
- Next Features
  - Secure authentication
    - Currently using username/email as login keys
    - Integrate Passport/Express
  - Authorized access to user console
  - Extended role creation
    - Currently limited to admin/user/guest