UberCarrot Documentation

Release 0.1a1

Callum McLean

December 14, 2015

Contents

UberCarrot is created from two different parts:

- The UberCarrot main project itself, which is primarily a *CasparCG* client that allows a user to use CasparCG as a graphics machine. Read more about that in All About UberCarrot.
- The backend to UberCarrot, a python module called `CasparTalk`. This provides a rough Python wrapper around the AMCP commands used by CasparCG, as well as a user- and developer- friendly way of implementing them. Read more at CasparTalk.

Contents:

CHAPTER 1

All About UberCarrot

UberCarrot *(working title)* is an interface that allows a user to use a CasparCG Server as a *graphics machine* or *character generator*.

This mostly happens by using Caspar/SVT AMCP commands over a TCP socket and working with Caspar Template and Dataset concepts.

1.1 Using UberCarrot

1.1.1 Creating Graphics On-The-Fly

For example, the process of taking a graphic (let's say a lower third) on air might be:

1. Open the graphic template (located on the CasparCG server) in UberCarrot.
2. The template will contain text fields that we can fill in. Use the text boxes in UberCarrot to fill these in.
3. Click the *Animate* button on the UberCarrot window. This will instruct CasparCG to take the graphic to air.
4. When we want the graphic to disappear from the output, clicking the *De-animate* button will instruct CasparCG to let the Flash template to play through the last section of its animation, which typically takes the graphic off the screen.

1.1.2 Creating Graphics In Advance

However, if we want, we can create the graphic ahead of time, and save it. Later, when we need it, we can recall the graphic (including the contents of the text-boxes) and play it out.

Warning: This workflow is not even close to confirmed, and may change later.

1. Choose a unique numeric ID (1-9999) for the graphic that you want to create. Type it in using the **numpad**. Press `Enter`.
2. Open the graphic template as above.
3. Fill in the fields provided.
4. Go to File -> Save, or, press `Ctrl-S`. The graphic (with associated text) has now been assigned to the ID number you chose.
5. When you want to recall the graphic, use the numpad to type the ID number and press `Enter`.
6. Animate and de-animate the graphic as normal.

CHAPTER 2

What Can UberCarrot Do?

Ultimately, at the moment: *not much*.

It's fair to say that this is still very much a work-in progress, and in the **alpha** stage of its lifecycle.

However, UC aspires to grand things! Here's what we *intend* to do (at the current stage):

- **Load CCG-generated Flash graphics templates**
  - Possibly some kind of media manager/template manager, to explore the contents of the Caspar server?
- Send arbitrary text strings to CCG in order to populate the graphics fields
- **Some form of dynamic text replacement?**
  - Maybe that means a lookup table of sorts, which refers to a plugin that returns a text string on every refresh?
  - Like `{{clock}}` refers to a function `make_clock` which returns a string
- A plugin architecture of sorts - but only of the field-replacement variety (as above) for now...
- **Ooh, maybe have a *preview window*!**
  - Figure out a way to have CCG generate a second output for the preview window dynamically
  - Or maybe have a Flash player panel, and see if we can get data into the Flash player...
- **Be able to save and recall graphics, pre-filled**
  - **Prepare graphics ahead of time, save them and load them back in when necessary**
    - This could be done by saving Caspar Datasets :)
- Shortcut keys - animate/de-animate?
- Is it in the remit of UC to be able to configure CCG?
- Graphics shotbox?
- Look pleasant

CHAPTER 3

CasparTalk

3.1 AMCP

3.1.1 Miscellaneous Functions

3.1.2 Query Commands

3.1.3 Data Commands

3.1.4 CG Commands

3.2 Caspar Objects

CHAPTER 4

Glossary

AMCP The *Advanced Media Control Protocol*, or AMCP, is *CasparCG*'s primary way of communicating with clients. Command strings are issued over a TCP socket to a given port on a CasparCG Server. CasparCG then parses the command and returns a string over the same socket, informing the user of the validity of the given command and any information that the user has requested. AMCP can be used to load, play and stop media on the server, manipulate datasets, and get or set information about the server itself.

CasparCG, CCG Open-source software developed by SVT, designed to play video content to a number of outputs, but can also manipulate the video's colour and spatial properties, as well as playing Flash- and HTML-based templates out, so as to be able to create keyed graphics. See CasparCG Server for more details.

Character generator, Graphics machine, CG Primarily used in TV, a character generator (CG) is a computer that creates on-screen graphics that can be keyed over an image. Some of the most common graphics that CGs create include lower-third straps (that might show the name of a TV presenter) or leaderboards in sports programmes.

SVT *Sveriges Television*, the Swedish national broadcasting organisation, and developers of CasparCG. They use CasparCG to control the output of their 6 TV channels.

UberCarrot, UC This project, but including CasparCG, which is separately maintained and developed by the Swedish national broadcaster, SVT.

CHAPTER 5

Indices and tables

- genindex
- modindex
- search