The Simulation and Neuroscience Application Platform (SNAP)

ftp://sccn.ucsd.edu/pub/SNAP

github.com/chkothe/SNAP

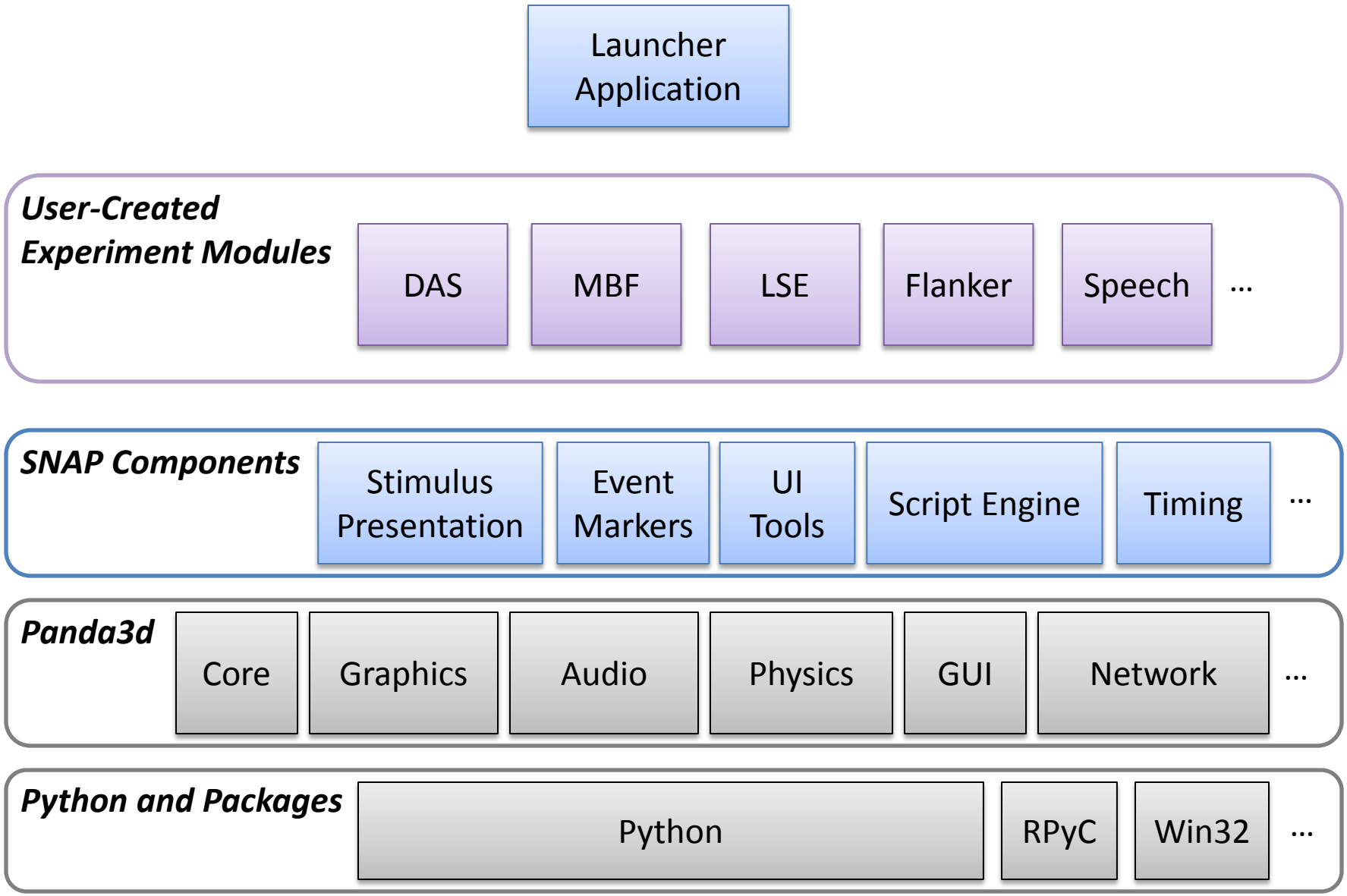
Purpose

- *Simple and easy-to-learn* scripting for basic neuroscience experiments
- *Scale to very complex* game-like human-computer interactions seamlessly
- Help *transition* basic neuroscience experiment paradigms into complex environments
- Full source code, *no license restrictions on academic or commercial use and deployment*

Approach

- Relies on **Python** as the scripting language and leverages its packages
- Uses the **Panda3d** game engine for graphics, audio, input, physics, GUI and low-level real-time subsystems
- Adds a thin **layer for experiment scripting**
- Adds some **extra low-level subsystems** (LSL, RPC, Pathfinding, …)

SNAP Architecture



Swartz Center for Computational Neuroscience

Basic Scripting

```
from framework.latentmodule import LatentModule
import random

class Main(LatentModule):
    def __init__(self):
        LatentModule.__init__(self)

        # set defaults for some configurable parameters:
        self.num_trials = 50        # number of trials in first part
        self.text_probability = 0.5 # probability that a text is displayed instead of a picture

    def run(self):
        self.marker(10)  # emit an event marker to indicate the beginning of the experiment
        self.write('This is a sample experiment.\nYou will be lead through a few trials in the f
        self.write('Press the space bar when you are ready.','space')

        for k in range(self.num_trials):
            # show a 3-second cross-hair
            self.crosshair(3)
            # display either a text or a picture
            if random.random() < self.text_probability:
                self.marker(1)
                self.write('A text.',scale=0.5)
            else:
                self.marker(2)
                self.picture('monkey.jpg',2,scale=0.3)
            # wait for 2 seconds
            self.sleep(2)

        self.sound('nice_bell.wav',volume=0.5)
        self.write('You successfully completed the experiment.')
```

Complex Scripting

- Example: earlier experiment prototype (MBF)

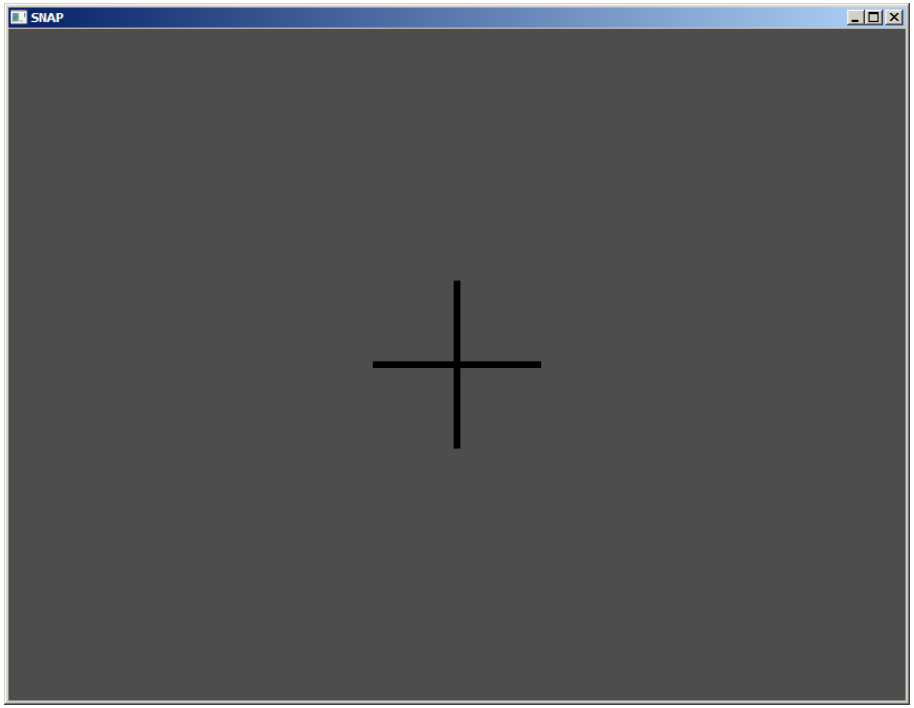
Integration with Other Tools

- Integrated with LSL (marker sending, remote control, real-time data access)
- Integrated with BCILAB (can use BCI signals and produce data for offline analysis) and others (EEGLAB, MoBILAB, SIFT)
- Open-ended design (Python, open-source, …)

Some Caveats

- Stimulus presentation not necessarily with same hard timing guarantees as traditional neuroscience applications (at least not in complex situations)
- Lacking rich authoring tools (e.g., dataflow graphs) of commercial software, relies fully on scripting and external authoring tools (Eclipse/PyCharm, 3dsmax/Maya, …)

Brief Demo



Thanks!

Questions?