VPython Architecture

Bruce Sherwood
Emeritus Professor, Department of Physics
North Carolina State University

Ruth Chabay
Professor, Department of Physics
High Point University

The development of VPython was funded in part by the National Science Foundation (grant DUE 0237132). Opinions expressed are those of the authors, and not necessarily those of the foundation.

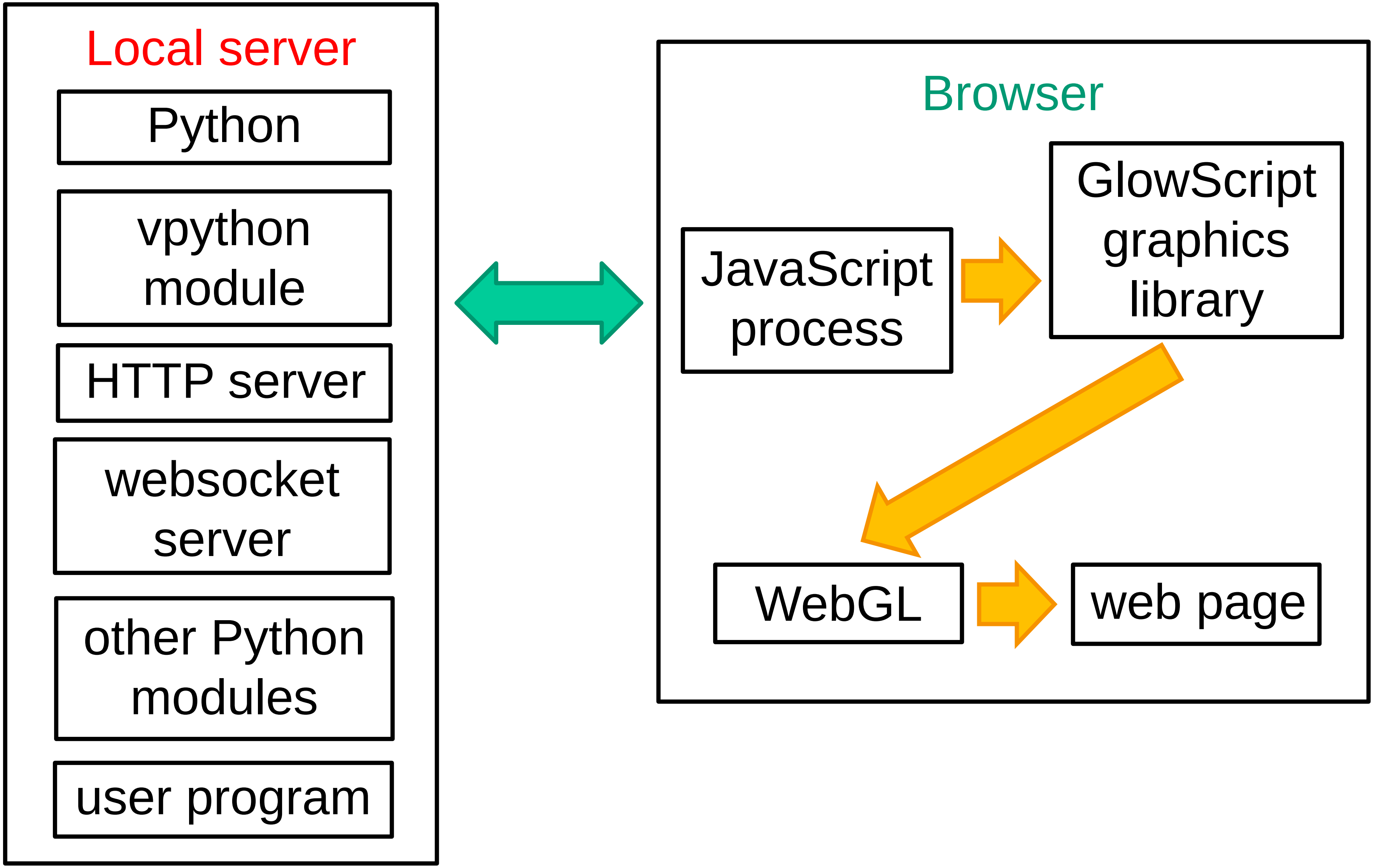
Empowering Nonexpert Programmers

- VPython: Python programming language plus 3D graphics
- Novice programmers can create navigable real-time 3D animations
- Main users: students, educators, researchers

Examples of VPython Programs

- A complex program can run in multiple environments
- A simple program
  - Intelligent defaults
  - Vector computations
  - 3D animations as side effects of computations
- Can embed in web pages

VPython 7



Local server

Python

vpython module

HTTP server

websocket server

other Python modules

user program

Browser

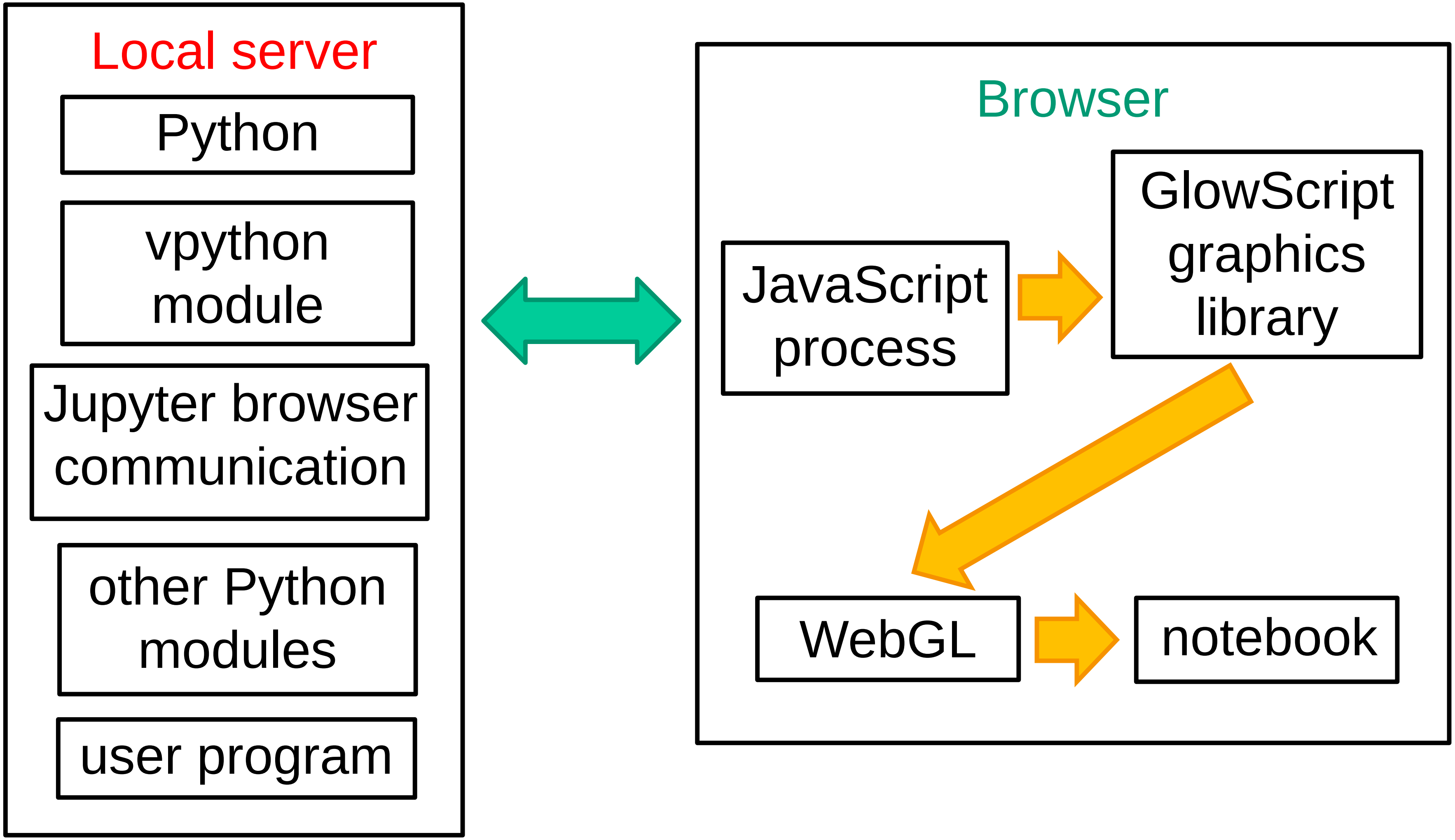
JavaScript process

GlowScript graphics library

WebGL

web page

VPython 7 with Jupyter Notebook



Local server

Python

vpython module

Jupyter browser communication

other Python modules

user program

Browser

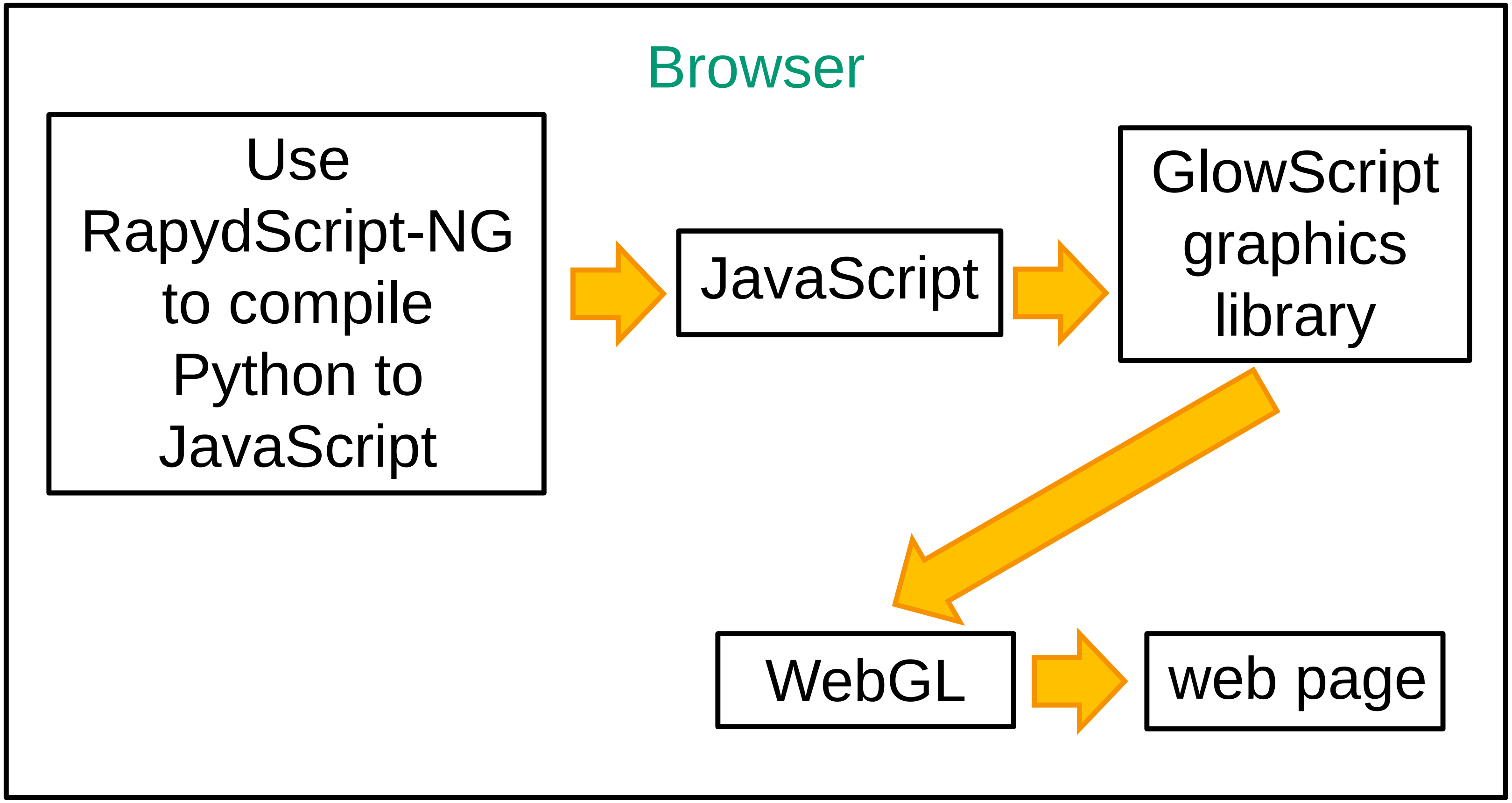
JavaScript process

GlowScript graphics library

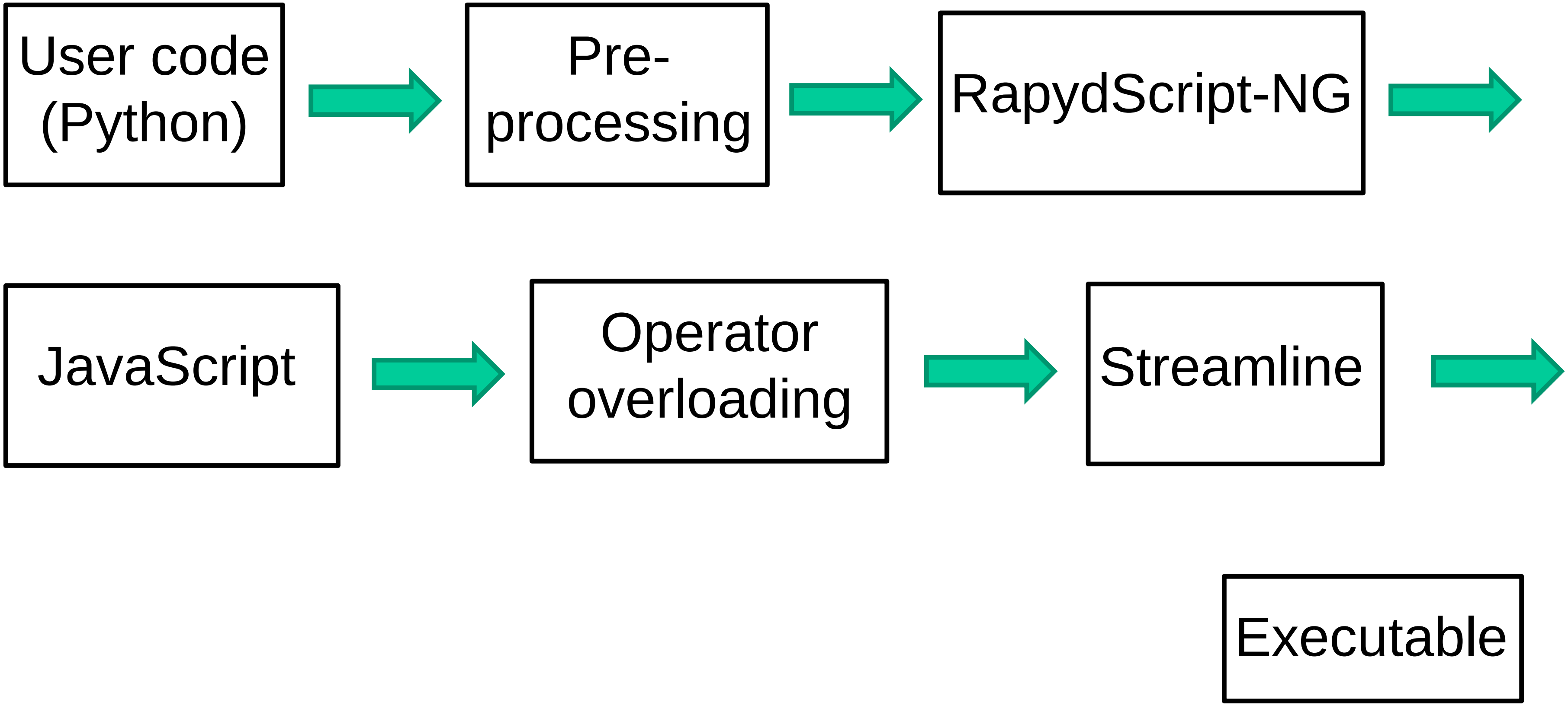
WebGL

notebook

GlowScript VPython
(all in browser)


Browser
Use RapydScript-NG to compile Python to JavaScript
JavaScript
GlowScript graphics library
WebGL
web page

GlowScript VPython: Compilation Details



User code (Python)

Pre-processing

RapydScript-NG

JavaScript

Operator overloading

Streamline

Executable

Operator Overloading: How Vector Addition Works

- Using the PaperScript library with the Acorn parser of JavaScript, convert

  ```
  a + b    =>    a[‘+’](b)
  ```

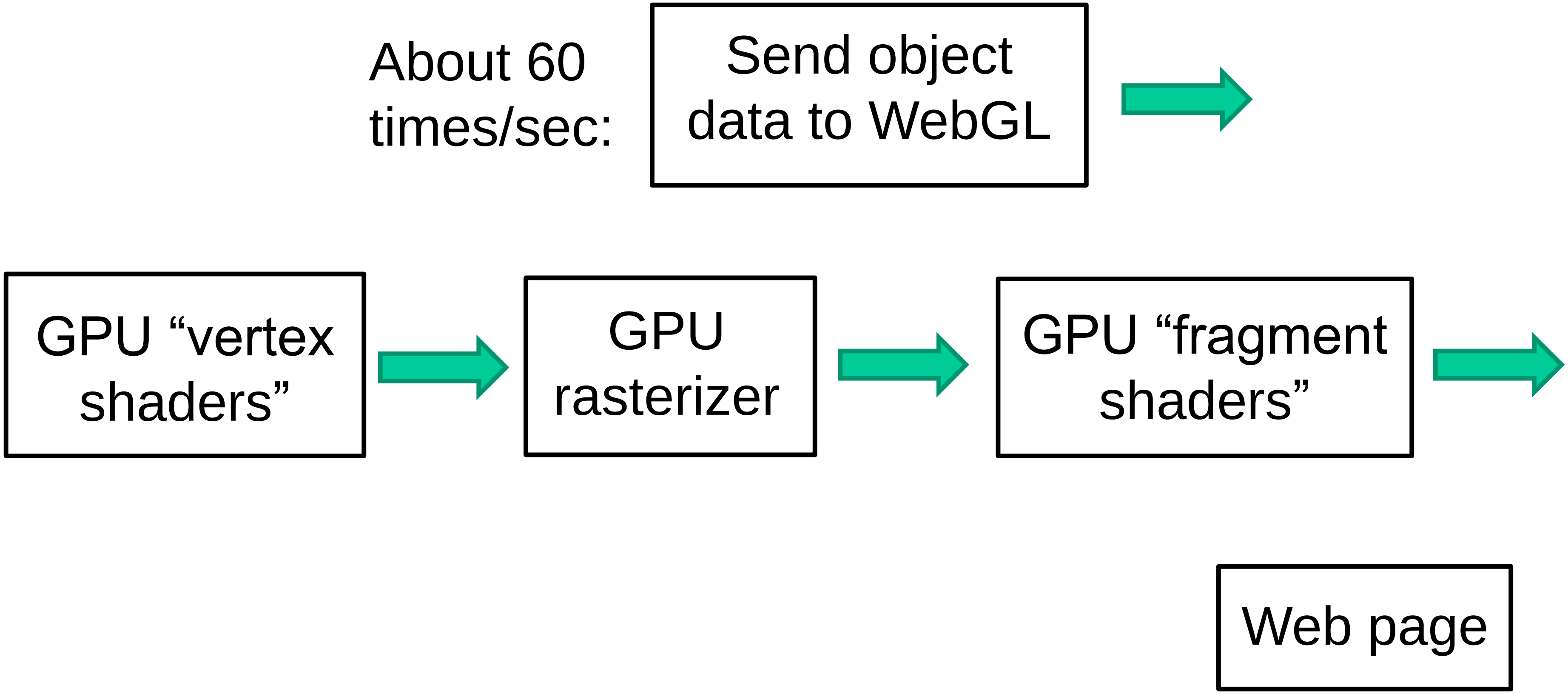
- JavaScript permits changing the behavior even of built-in classes such as Number and String

-
  ```
  String.prototype['+'] = function(r) { return this + r }
  ```

-
  ```
  Number.prototype['+'] = function(r) {
     return (r instanceof vec) ? add_error() : this + r }
  ```

-
  ```
  vec.prototype['+'] = function(r) {
     return (r instanceof vec) ?
       new vec(this.x + v.x, this.y + v.y, this.z + v.z) add_error() }
  ```

GlowScript Rendering of 3D images



About 60 times/sec:

Send object data to WebGL

GPU “vertex shaders”

GPU rasterizer

GPU “fragment shaders”

Web page

Animation Loop

- rate(200): no more than 200 loop iterations/s
- About 60 renders/s
- Sleep for remaining time
- Assigning to an object attribute (pos, size, etc.) sets a “changed” flag for that object, and at render time its current attributes are repackaged to send to GPU
- Five 4-vectors: pos, axis, size, color, up, with texture, opacity, shininess, emissive packed into 4th slots; total of 80 bytes per object instance

Object Models in GPU Memory

- A “model” box object is stored in GPU memory
- Represented by 12 triangles, each described by 3 vertex objects specifying position, normal, color, and texture coordinates
- Data for a particular box (an instance of the box class) plus model information is sufficient for the GPU to display that box appropriately in 3D
- GPU memory has models of a box, sphere, cylinder, cone, and pyramid; compounds treated like primitives
- Arbitrary objects built from triangles; extrusions, 3D text

Speed Issues

- Python is an interpreted language and so execution is significantly slower than compiled languages.
- Computationally intensive GlowScript VPython programs run about an order of magnitude faster than VPython 7 programs, because they are compiled to (fast) JavaScript (but there is no access to Python modules).

Additional Technical Details

- Portions of objects hidden behind other objects are not seen thanks to “z-depth” blocking by GPU hardware
- Transparency handled by “depth peeling” algorithm
- Mouse “picking” uses false colors

Major Contributors to GlowScript VPython and VPython 7

- David Scherer: originator of VPython; major contributions to the start of the GlowScript project
- John Coady: originator of Jupyter VPython
- Matt Craig: installers for VPython 7

Brief History

- 2000: Classic VPython created by David Scherer, an undergraduate student at Carnegie Mellon University, in collaboration with Chabay and Sherwood
- 2011 GlowScript begun by Scherer and Sherwood
- 2014 GlowScript VPython by Sherwood
- 2015 Jupyter VPython begun by John Coady
- 2016 VPython 7: Jupyter VPython made consistent with GlowScript VPython by Chabay and Sherwood, in collaboration with Coady; Classic VPython no longer supported

See **brucesherwood.net** for a detailed history

For More Information

- **vpython.org** – obtaining and using VPython
- **glowscript.org** – full VPython documentation, many examples
- **trinket.io** – embed both editing and execution of VPython in your own web page
- **matterandinteractions.org** – calculus-based contemporary intro physics curriculum in which VPython plays an important role
- **matterandinteractions.org/student** – includes a large number of physics demo programs written in VPython