The Sage Project's Primary Goal

Create a viable free open source alternative to Magma, Maple, Mathematica, and Matlab.

Firefox  <-->  Internet Explorer, Opera
Open Office, Latex  <-->  Microsoft Office
Linux  <-->  Microsoft Windows
PostgreSQL, MySQL  <-->  Oracle, Microsoft SQLserver
GIMP  <-->  Photoshop
Sage  <-->  Magma, Maple, Mathematica, Matlab
Brief History of the Sage Project


- Sage-1.0 released February 2006 at Sage Days 1 (UC San Diego).

- 20 Sage Days Workshops (!) at UCLA, UW, Cambridge, Bristol, Austin, France, San Diego, Seattle, MSRI, ..., Barcelona (next week at UPC!), Lopez Island

- Sage won first prize in the Trophees du Libre (November 2007)

- Funding from Microsoft, Univ of Washington, UC San Diego, NSF, DoD, Google, Sun, private donations, etc.
Sage's @interact -- image compression example

Use singular value decomposition to compress the courtyard outside.

```python
import pylab
A_image = pylab.mean(pylab.imread('ub.png'), 2)
@interact
def svd_image(i = ('Eigenvalues (quality)', (20, (1..100))),
             display_axes = ('Display Axes', True)):
    u, s, v = pylab.linalg.svd(A_image)
    A = sum(s[j]*pylab.outer(u[0:,j], v[j,0:]) for j in range(i))
    g = graphics_array([[matrix_plot(A), matrix_plot(A_image)])
    show(g, axes=display_axes, figsize=(8,3))
    html('<h2>Image compressed using %s eigenvalues</h2>' % i)
```

Eigenvalues (quality)  17
Display Axes  

Image compressed using 17 eigenvalues
Sage -- An Open & International Development Effort

1. Over 150 contributors total -- see the developer map.
2. Copious credit given to every developer's contributions in every release
3. New stable release every 2-3 weeks
4. Rotating group of release managers
5. All bugs etc. publicly tracked at http://trac.sagemath.org
Summary:
Sage is about building the car instead of reinventing the wheel

1. Sage uses a *popular mainstream programming language* instead of inventing a custom mathematics language.

2. Use straightforward method to link programs together -- *C library and pseudotty's*, instead of XML servers/OpenMath. *We implement all conversion routines, instead of expecting upstream to do it: we make them communicate with Sage, whether they want to or not. Resistance is futile.*

3. *Give copious credit to contributors* and be very developer friendly (easily build from source).

4. Reuse, improve, and *contribute to existing libraries and projects* (e.g., Singular, Linbox, NTL, Pari, GAP, Maxima), instead of starting over and competing with them.

5. Make the GUI using a web browser: the world of java and javascript plugin is immediately available and Sage *integrates with the web.*

»Every free computer algebra system I’ve tried has reinvented many times the wheel without being able to build the car.«