

CS1110

Lecture 6: Function calls

Announcements

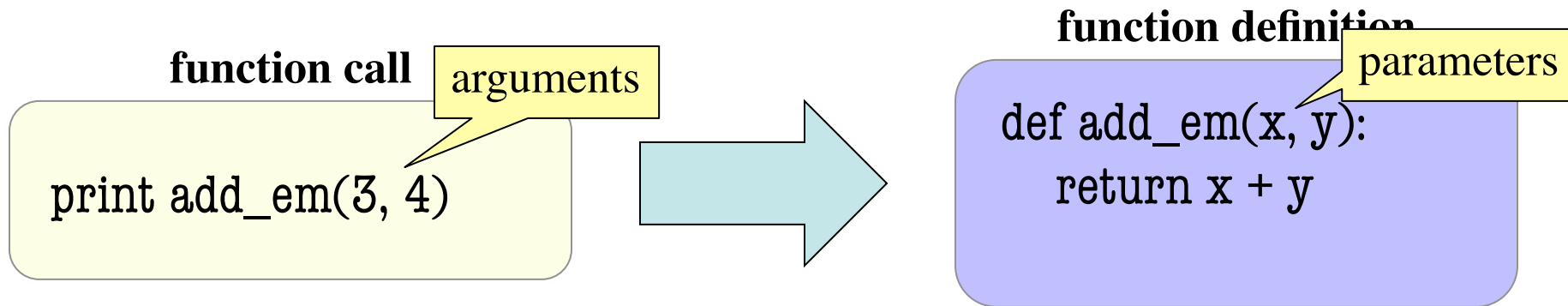
Additional space in labs:

We have added some space and staffing to the 12:20 and 1:25 labs on Tuesday. There is still space to move into these labs.

Assignment 1 is out! Grab a printed copy today, and refer to the same text on the website. This assignment is due **February 18**, and you will re-submit until it's all correct.

Printed copies are coming in a few minutes!

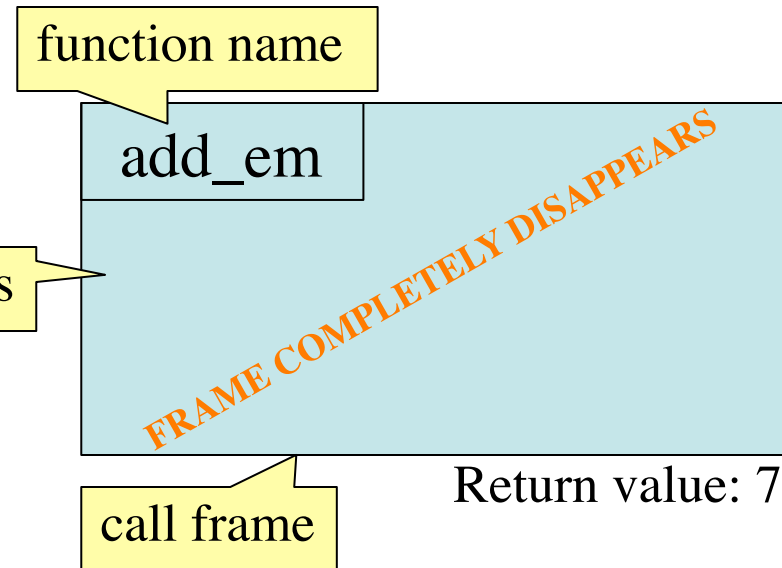
How Do Functions Work?



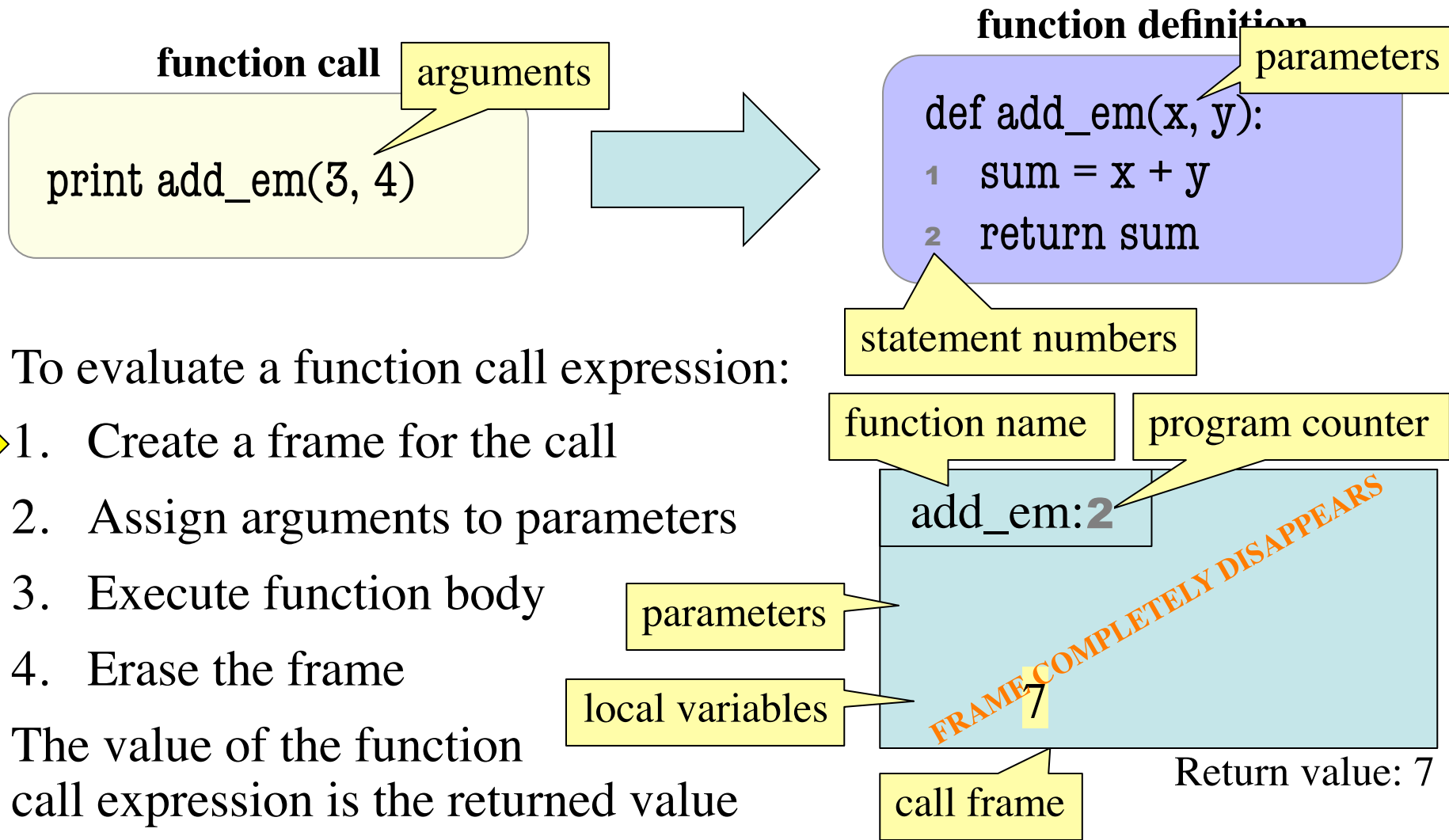
To evaluate a function call expression:

1. Create a frame for the call
2. Assign arguments to parameters
3. Execute function body
4. Erase the frame

The value of the function call expression is the returned value



How Do Functions Work?



function call

```
a = 1  
b = 2  
swap(a, b)
```

function definition

```
def swap(x, y):  
1  x = y  
2  y = x  
3  print 'x, y:', x, y
```

with print, use
commas to print
several values

To evaluate a function call expression

1. Create a frame for the call
2. Assign arguments to parameters
3. Execute function body
4. Erase the frame

The value of the function
call expression is the returned value

Execute this call on paper.
What gets printed out?

- A: x, y: 1 2
- B: x, y: 2 1
- C: x, y: 2 2
- D: x, y: 1 1
- E: I don't know

function call

```
a = 1  
b = 2  
swap(a, b)
```

arguments

module (global)
variables

a 1

b 2

function definition

```
def swap(x, y):  
1 x = y  
2 y = x  
3 print 'x, y:', x, y
```

parameters

To evaluate a function call expression:

1. Create a frame for the call
2. Assign arguments to parameters
3. Execute function body
4. Erase the frame

The value of the function call expression is the returned value

program counter

swap: 1

parameters

x

y

function call

```
a = 1
b = 2
swap(a, b)
print 'a, b:', a, b
```

arguments

module (global)
variables

a 1

b 2

function definition

```
def swap(x, y):
1  t = x
2  x = y
3  y = t
4  print 'x, y:', x, y
```

parameters

To evaluate a function call expression:

1. Create a frame for the call
2. Assign arguments to parameters
3. Execute function body
4. Erase the frame

The value of the function call expression is the returned value

program counter

swap: 1

parameters

x

y

local variables

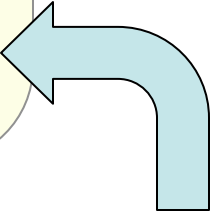
t

function call

```
import point
p = point.Point(1,2,3)
q = point.Point(3,4,5)
swap_x(p, q)
print 'p:', p
print 'q:', q
```

function definition

```
def swap_x(p, q):
  1 t = p.x
  2 p.x = q.x
  3 q.x = t
```



Execute this code on paper.
You will draw 2 objects and a frame.
What is in p.x at the end?

- A: 1
- B: 2
- C: 3
- D: I don't know

function call

```
import point
p = point.Point(1,2,3)
q = point.Point(3,4,5)
swap_x(p, q)
print 'p:', p
print 'q:', q
```

module (global)
variables

p **id1**

q **id2**

function definition

```
def swap_x(p, q):
  1 t = p.x
  2 p.x = q.x
  3 q.x = t
```

id1

Point

x 1.0

y 2.0

z 3.0

id2

Point

x 3.0

y 4.0

z 5.0

swap_x:1

p

t

q

(9:05 version)

function call

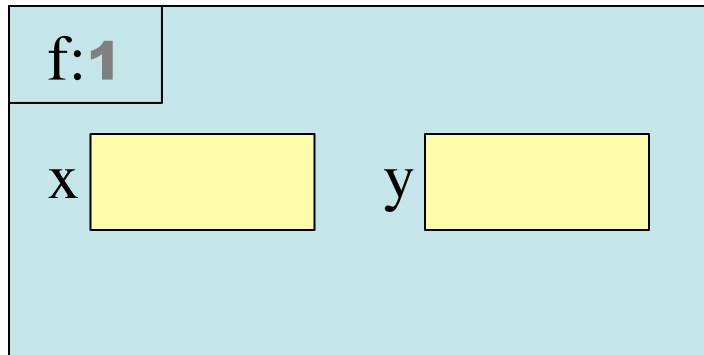
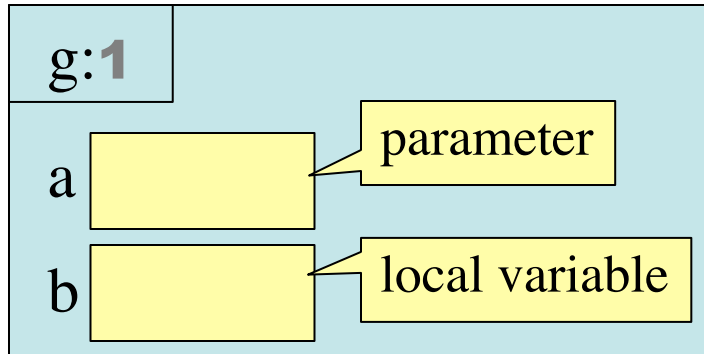
```
c = 2  
print g(3)
```

c 2

function definitions

```
def f(x, y):  
1 return 3*x + y
```

```
def g(a):  
1 b = f(a, c)  
2 return f(b, a)
```



1. Create a frame for the call
2. Assign arguments to parameters
3. Execute function body
4. Erase the frame

(11:15 version)

function call

```
lt_speed = 3e8  
print g(3)
```

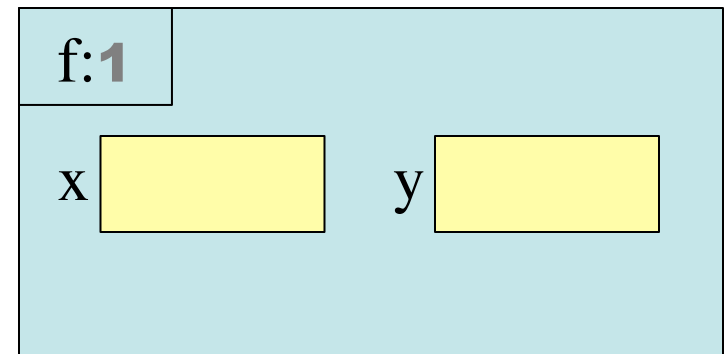
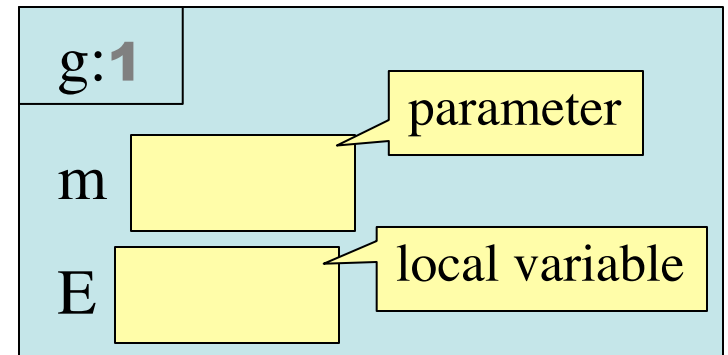
function definitions

```
def g(m):  
1 E = f(m, lt_speed)  
2 return E
```

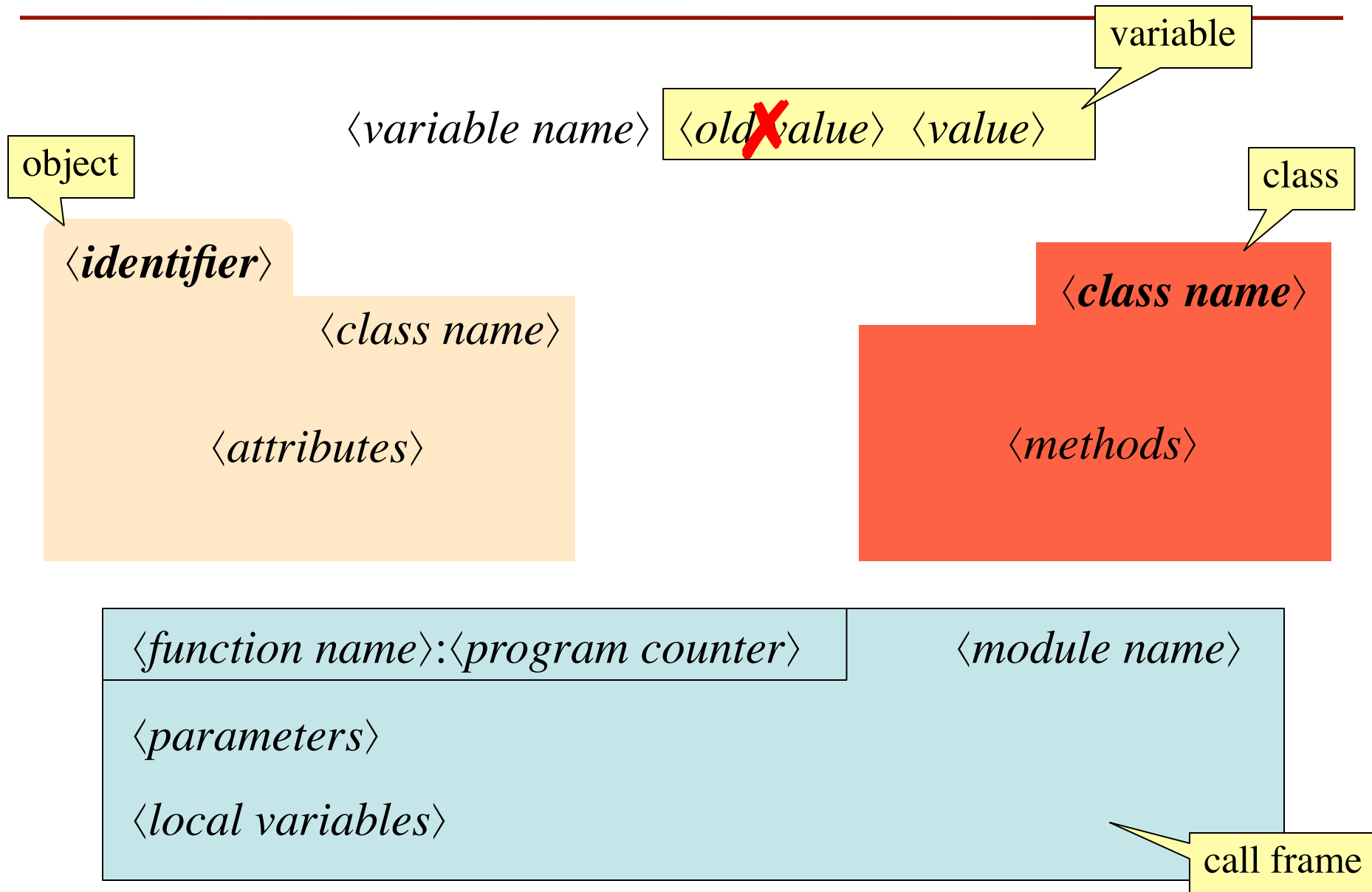
```
def f(x, y):  
1 return x * (y**2)
```

lt_speed 3×10^8

1. Create a frame for the call
2. Assign arguments to parameters
3. Execute function body
4. Erase the frame



How to Draw Things



Online Python Tutor

pythontutor.com

type in whatever code you want

controls for stepping through code

module (global) variables

frame for call of g

frame for call of f

output from print goes here

The screenshot displays the Online Python Tutor interface. The code editor contains the following Python code:

```
1 c = 2
2
3 def f(x, y):
4     return 3*x + y
5
6 def g(a):
7     b = f(a, c)
8     return f(a, b)
9
10 print g(3)
```

The execution progress is shown as "Step 10 of 11". The "Global variables" table is as follows:

Variable	Value
c	2
f	id1
g	id2

The "Objects" table is as follows:

Object ID	Object
id1	function f(x, y)
id2	function g(a)

The "Frames" table shows the current call stack:

Frame	Variables
g	a: 3, b: 11
f	x: 3, y: 11, Return value: 20

The "Program output" section is currently empty.

At the bottom, there is a "Generate URL" button and a text box for sharing the visualization.

To share this visualization, click the 'Generate URL' button above and share that URL. To report a bug, paste the URL along with a brief error description in an email addressed to philip@pgbovine.net