

Groningen Graduate School of Humanities' "Interchange" presents:

Deep Meaning Annotation Project

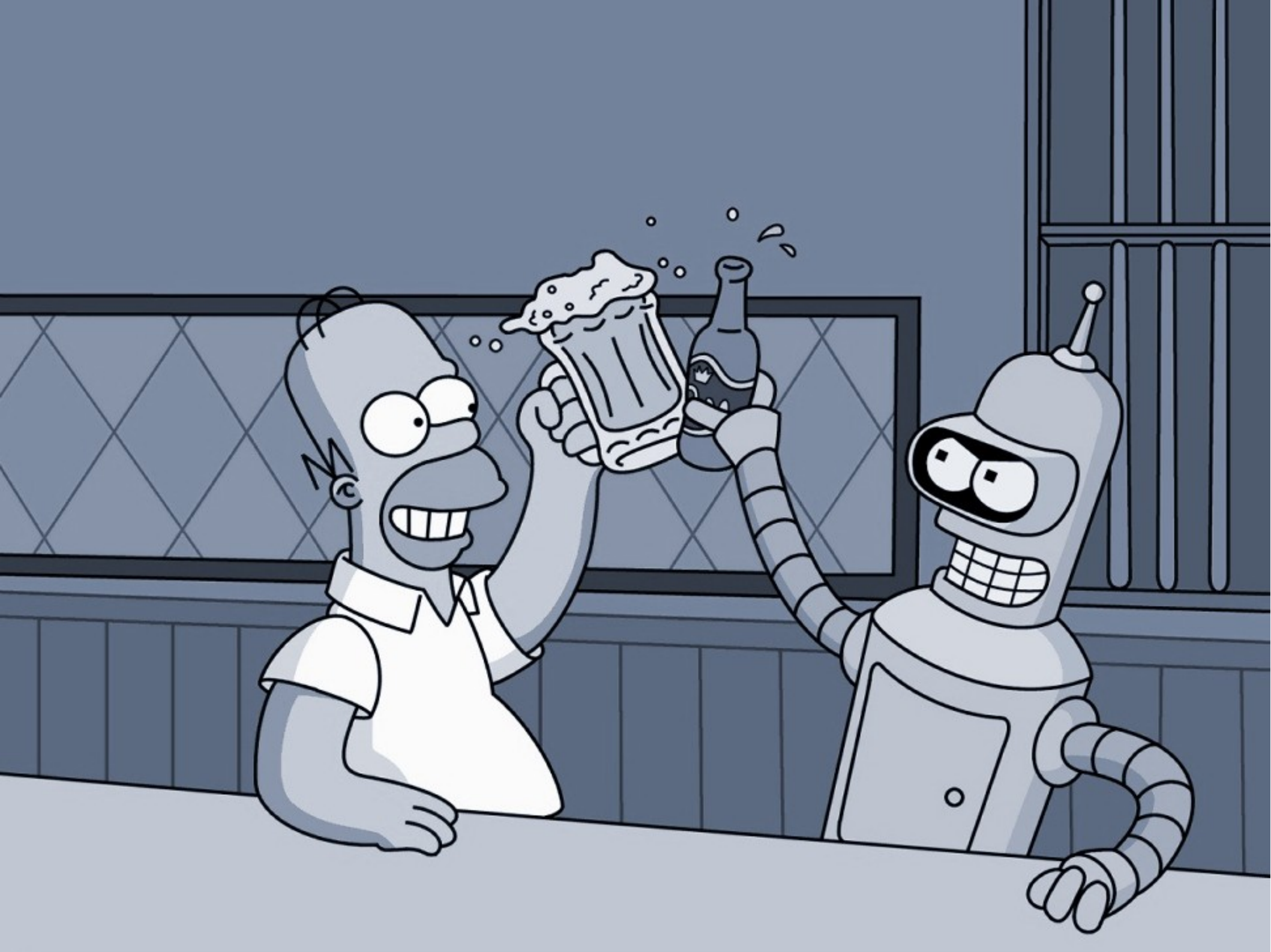
Valerio Basile

31/5/2013

**Communication
is key**

**Communication
is key**

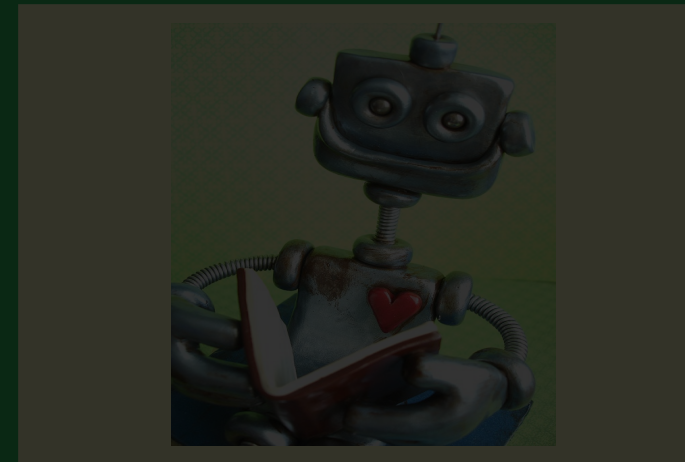




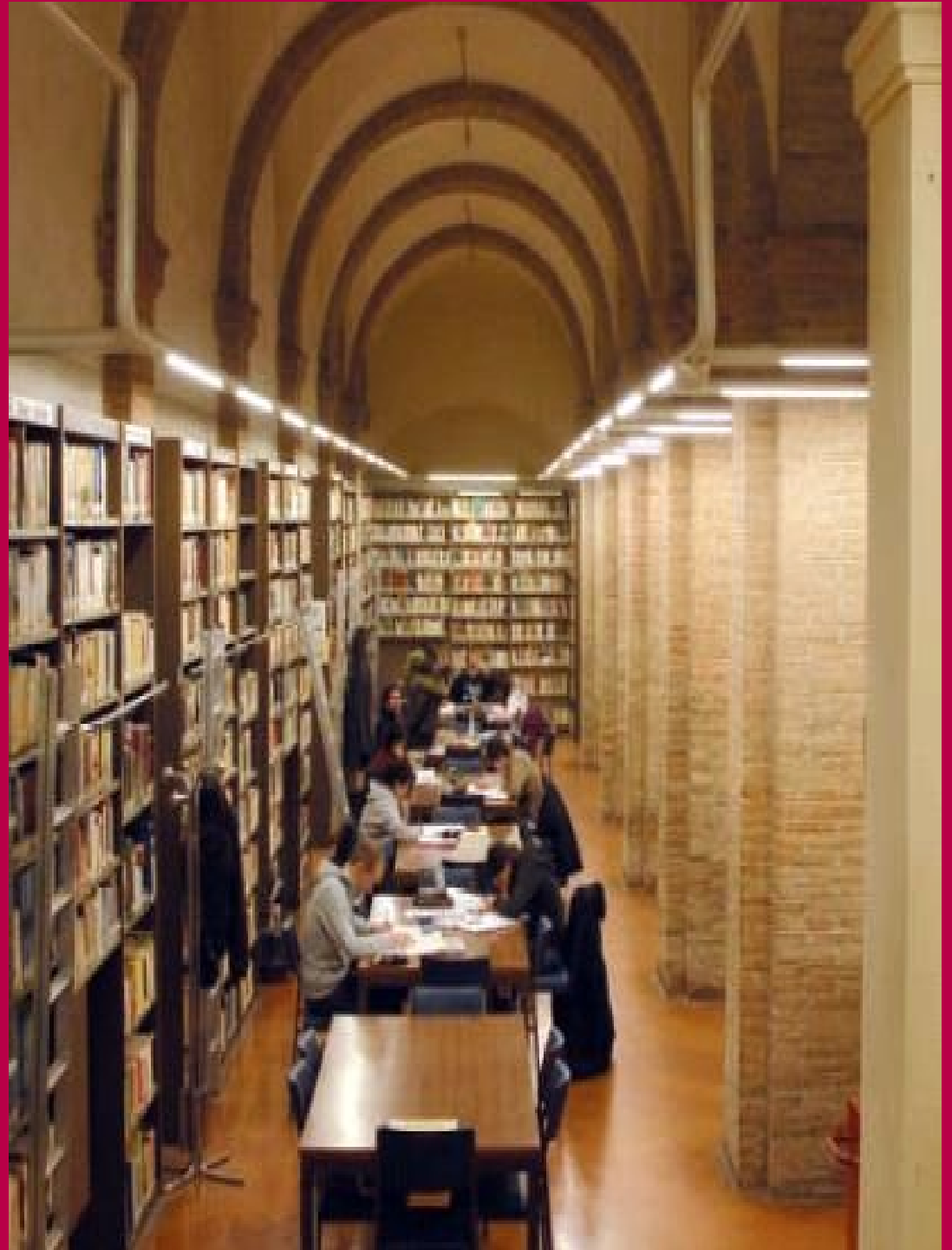
Ceci n'est pas une pipe



```
1 #!/usr/bin/perl
2
3 # This script demonstrates the use of the Perl module
4 # "Data::Dumper" to dump the contents of a hash
5 # and an array.
6
7 use Data::Dumper;
8
9 # Create a hash and an array
10 my %hash = (
11     'name' => 'John Doe',
12     'age' => 30,
13     'city' => 'New York'
14 );
15 my @array = (1, 2, 3, 4, 5);
16
17 # Dump the hash and array
18 print Dumper \%hash;
19 print Dumper \@array;
20
21 # The output of the script will be:
22 # {
23 #   "age" => 30,
24 #   "city" => "New York",
25 #   "name" => "John Doe"
26 # }
27 # [1, 2, 3, 4, 5]
```



Text,
a lot
of text





Ceci n'est pas une pipe



Ceci n'est pas une pipe



Ceci n'est pas une pipe



Ceci n'est pas une pipe

19

→



Ceci n'est pas une pipe

19



Ceci n'est pas une pipe

10 ♀

(The text is written in black cursive with red hand-drawn circles around 'Ceci', 'une', and 'pipe', and red arrows pointing from the text to the pipe above.)

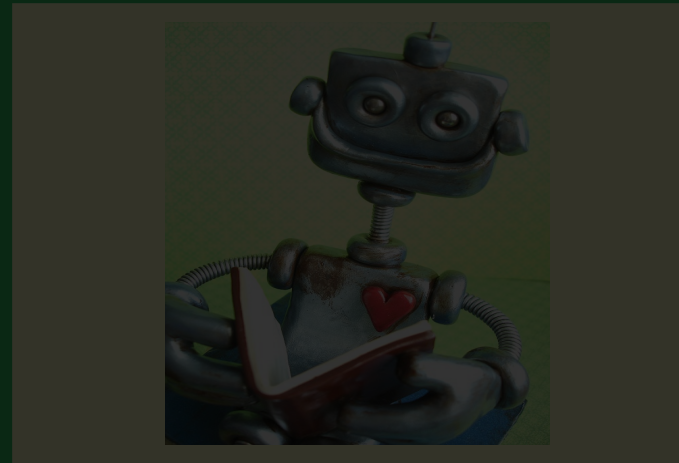


<http://gmb.let.rug.nl>

Ceci n'est pas une pipe



```
1 1
2 2
3 3
4 4
5 5
6 6
7 7
8 8
9 9
10 10
11 11
12 12
13 13
14 14
15 15
16 16
17 17
18 18
19 19
20 20
21 21
22 22
23 23
24 24
25 25
26 26
27 27
28 28
29 29
30 30
31 31
32 32
33 33
34 34
35 35
36 36
37 37
38 38
39 39
40 40
41 41
42 42
43 43
44 44
45 45
46 46
47 47
48 48
49 49
50 50
51 51
52 52
53 53
54 54
55 55
56 56
57 57
58 58
59 59
60 60
61 61
62 62
63 63
64 64
65 65
66 66
67 67
68 68
69 69
70 70
71 71
72 72
73 73
74 74
75 75
76 76
77 77
78 78
79 79
80 80
81 81
82 82
83 83
84 84
85 85
86 86
87 87
88 88
89 89
90 90
91 91
92 92
93 93
94 94
95 95
96 96
97 97
98 98
99 99
100 100
```





[Help](#) |

Google Image Labeler

time left

01:47

score

0

passes

0

label

pass

Your partner has suggested 4 labels.

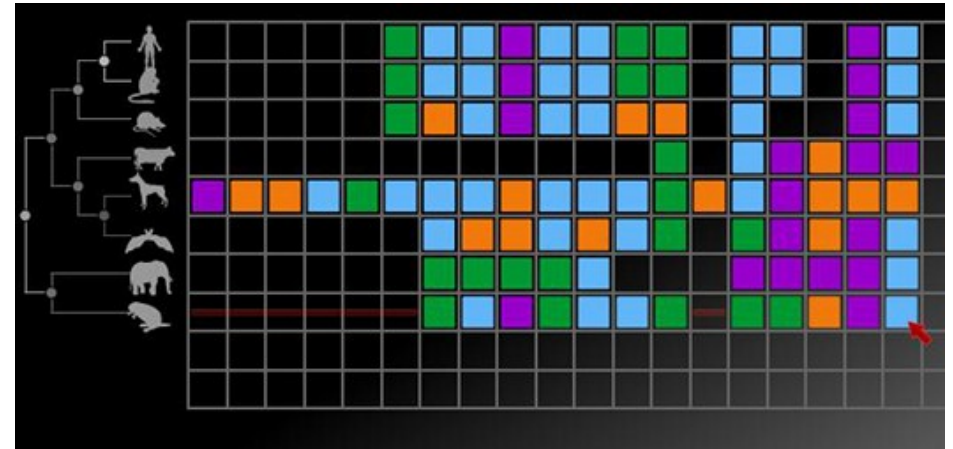


zoom out

off-limits

spaghetti
pasta
food
green
noodles

my labels



Google Image Labeler

[Help](#) |

time left

01:47

score

0

passes

0

label

pass

Your partner has suggested 4 labels.

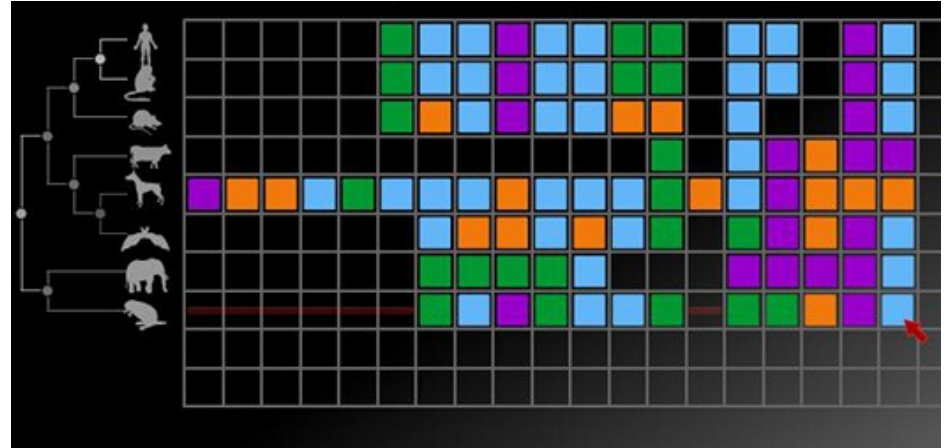


zoom out

off-limits

spaghetti
pasta
food
green
noodles

my labels



Google Image Labeler

[Help](#) |

time left

01:47

score

0

passes

0

label

pass

Your partner has suggested 4 labels.



zoom out

off-limits

spaghetti
pasta
food
green
noodles

my labels



explorer

Ukraine

NNP ▾

Location ▾

- O
- Person
- Organization
- Location
- GPE
- Artifact
- Event
- Natural_Object
- Time

xv0. (x2

named(x2, ukraine, loc)

says

VBZ ▾

O ▾

1: state say tell ▾

$\lambda v0. \lambda v1. \lambda v2. (v1 @ \lambda v3. (e5 p6 t8 t9 ; (v2 @ e5)))$

now(t8)
e5 \subseteq t9
t9 = t8

Himalaya, Amazon Rainforest, Atlantic Ocean, Rhine, Venus, Milky Way, 742 Evergreen Terrace Springfield, <http://gmb.let.rug.nl>, chunkylover53@aol.com, 0629555697...

?

wordrobe


play what you mean

 Senses Questions left until drawer is completed: 5



The Who is currently touring in support of *Endless Wire*, its first **album** since 1982.

- one or more recordings issued together – originally released on 12-inch phonograph records (usually with attractive record covers) and later on cassette audiotape and compact disc
- a book of blank pages with pockets or envelopes – for organizing photographs or stamp collections etc

Place your bet: low  high

answer

skip

Ceci n'est pas une pipe



```
#!/usr/bin/perl

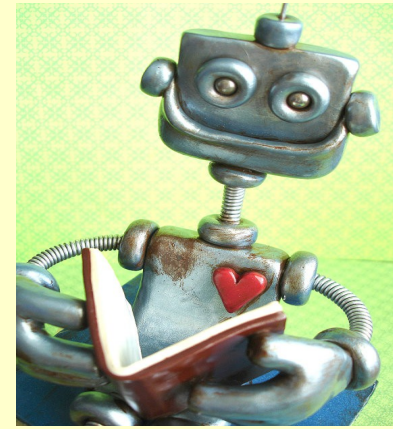
use strict;
use warnings;

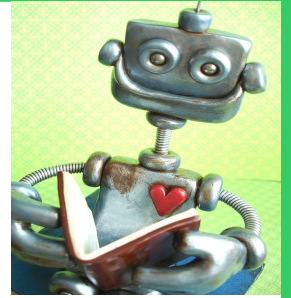
my $pipe = "perl -e 'print \"Ceci n'est pas une pipe\"'";

system($pipe);

my $output = `perl -e 'print \"Ceci n'est pas une pipe\"'`;

print $output;
```

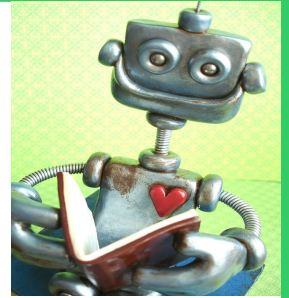




Sedimentary **rock** is formed in layers



Sedimentary **rock** is formed in layers



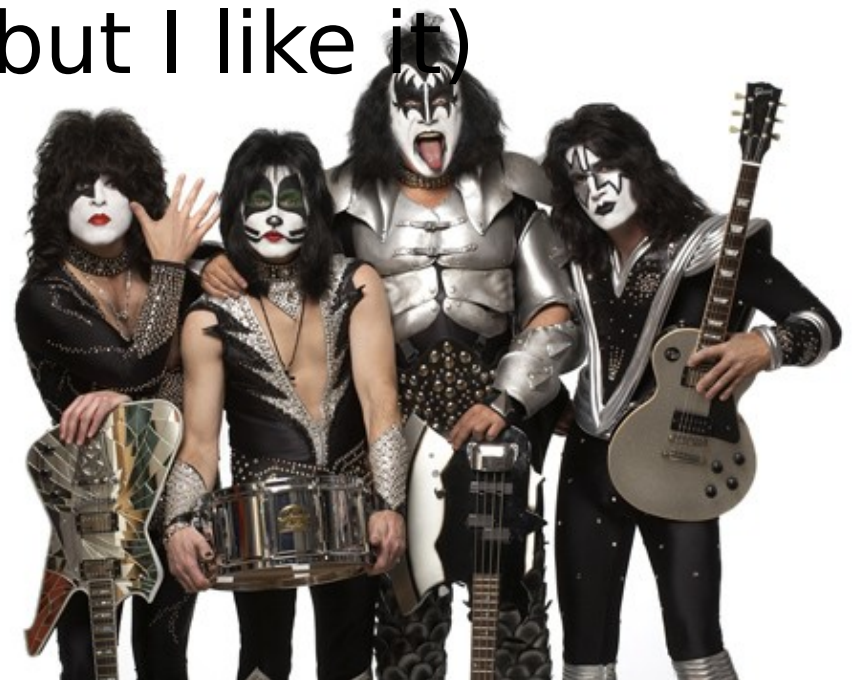
Sedimentary **rock** is formed in layers

It's only **rock** and roll (but I like it)



Sedimentary **rock** is formed in layers

It's only **rock** and roll (but I like it)



Ceci n'est pas une pipe



```
class Class1 {
    /// <summary>
    /// The main entry point for the application.
    /// </summary>
    [STAThread]
    static void Main(string[] args) {
        // Create
        BTULicenseManager licenseManager = new BTULicenseManager();
        // Get some valid license keys
        string networkLicense = "XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXXXXXXXXXX";
        string password = "12345678";
        licenseManager.Login(networkLicense, password);
        Console.WriteLine("Logged on.");
        string fullName = @"C:\Documents and Settings\kron.DMP\FREEM-My Videos\South Park-Freaky Stripes-2004-08-17-0.mpg";
        BTULibrary library = new BTULibrary();
        // Get properties
        PUSPropertyObj bag = library.GetMediaByFullName(fullName);
        // Print properties to the console
        Console.WriteLine("Properties of {0}", fullName);
        foreach (PUSProperty prop in bag.Properties) {
            Console.WriteLine("Property: {0}, {1}", prop.Name, prop.Value);
        }
        // For the PUSPropertyObj there is a new friendly collection class
        // It's a good idea for you to write a friendlier wrapper class that
        // will allow you to add and remove properties and ease back up.
        // See the following link for the details
        // http://www.muhimbi.com/Articles/PUSPropertyObj.aspx
        // Create a new PUSPropertyObj with the edited property
        PUSPropertyObj newBag = new PUSPropertyObj(bag);
        newBag.Properties = (PUSPropertyObj[])new PUSProperty[] {
            new PUSProperty("TitleDescription", "The boys decide to appear on a talk show, Edited by Beyond 10 Freemanb")
        };
        // Create a new PUSPropertyObj with the edited property
        PUSPropertyObj newBag2 = new PUSPropertyObj(newBag);
        newBag2.Properties = (PUSPropertyObj[])new PUSProperty[] {
            new PUSProperty("TitleDescription", "The boys decide to appear on a talk show, Edited by Beyond 10 Freemanb")
        };
        // Print properties to the console and verify the changes
        Console.WriteLine("Edited properties of {0}", fullName);
        foreach (PUSProperty prop in newBag2.Properties) {
            Console.WriteLine("Property: {0}, {1}", prop.Name, prop.Value);
        }
        // Print the new properties to the console and verify the changes
        Console.WriteLine("Press any key to exit...");
        Console.ReadKey();
        return;
    }
}
```



