Slides:

Pros

1. Pattern Matching
2. Inserting variables into a string
3. Compile times/speed
4. Example of a module 🡪 creating module
   1. Useful for OO programming, providing reusable code/reusable library
5. Template modules 🡪provide a list of them
6. Slide with stats on usage/authors
7. Do more by writing less code compared to other languages 🡪 i.e. change variable and change it in one line: ($b =$a) =~s/one/two/g;

* What this does is assign $a to $b then change “one” to “two” in $b
* $a = “one one two”;
* $b = “two two two”;
* OR ex. on sorting hash key values
* On same slide incorporate 2 ways to do the same thing 🡪 cgi module i.e. functional vs. OO way to do the same thing. i.e. $response->param(“y”) vs param(“y”)

1. Don’t have to declare datatypes 🡪 variables are context dependant i.e. $a==$b is a numeric comparison vs $a eq $b as strings (both versions stored). Probably first language to do that.

Cons

1. Problematic deployment of applications because of installing CPAN modules
2. Leaning curve and learning to install libraries/configure system
3. Mod\_perl – code example and description
4. FastCGI, – code example and description
5. http/cgi– code example and description
6. unoriginal syntax/possibly a history slide

Framework (8 slides)

1. Catalyst
2. CGI::Application

Future

1. Perl 6 🡪 not backwards compatible

Intro

* Perl came out in 1999 due to shell scripting inadequacies