Course 1: UNIX and C

Kristaps Dzonsons

01 December, 2011

Course site: *http://kristaps.bsd.lv/minicourse_12_2011*

Environment

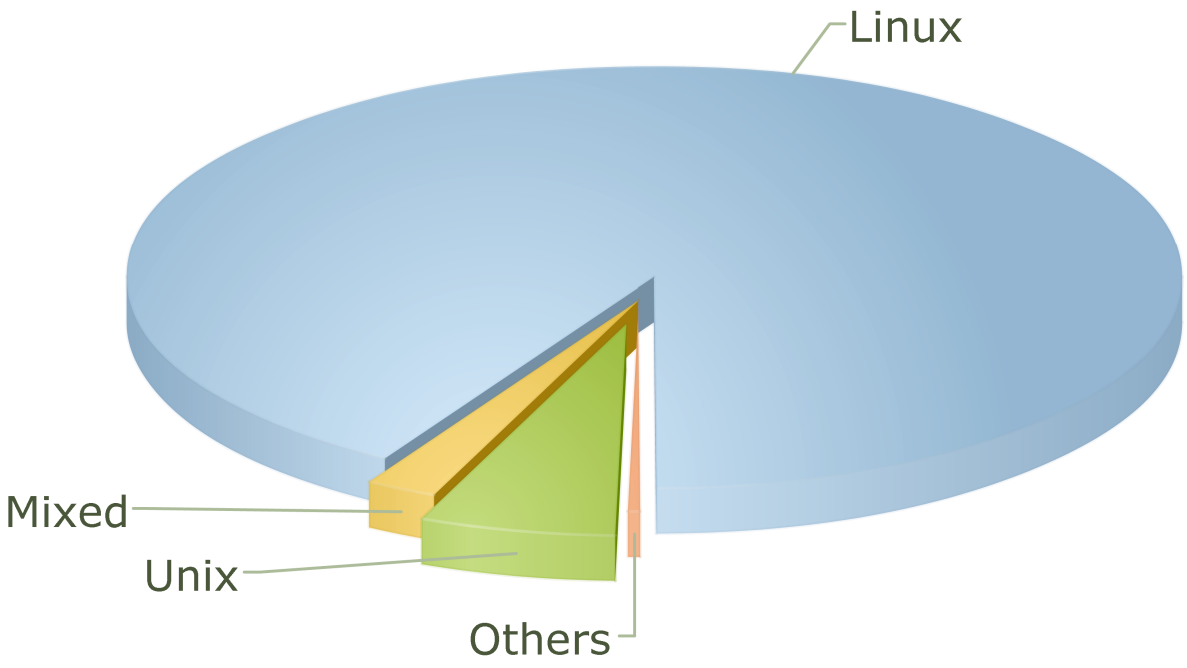
Most high-performance computing environments consist of:

- **UNIX** an operating system popular in non-desktop environments
- **C** a low-level, minimal programming language
- **hardware** purpose-built, high-performance hardware

In this lecture, we focus on *UNIX* and *C*.

UNIX: Why?

UNIX in the Top 500 Supercomputers: 99.8%



Source: *top500.org*, November, 2011

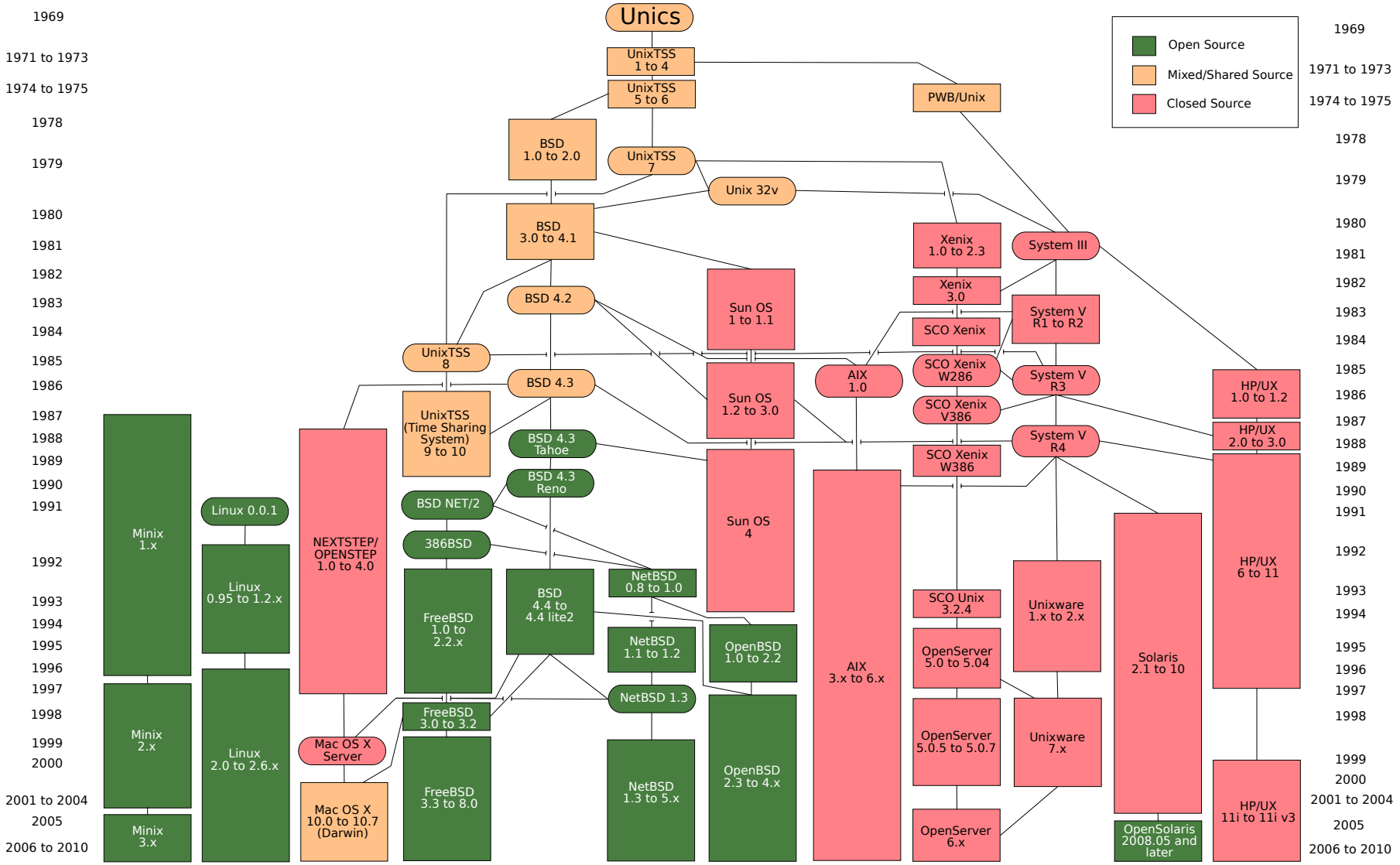
UNIX: Why?

UNIX on Swedish Supercomputers (KTH/PDC)

Lindgren	Cray XE6 (June 2011, 31/500 ranking): Linux
Ekman	Dell PowerEdge: Linux
Povel	Prototype Cluster: Linux
Zorn	NVIDIA GPU Cluster: Linux
Hebb	IBM Blue Gene/L: Linux

Source: *www.pdc.kth.se*

UNIX: What is it?



Source: commons.wikimedia.org

UNIX: What is it?

```
login: guest
Password:
$ help
```

UNIX: How do I use it?

Many resources exist to learn UNIX.

Web *www.ee.surrey.ac.uk/Teaching/Unix/*

...

Books Brian Kernighan and Rob Pike, *The UNIX Programming Environment*, Prentice Hall, 1984.
Arnold Robbins, *UNIX in a Nutshell*, O'Reilly Media, 2005.

...

When you first log in, run the following:

```
$ help
```

UNIX: Logistics

HHS officially does not yet have a UNIX server for computing.

HHS unofficially does have a UNIX server for experimenting and testing.

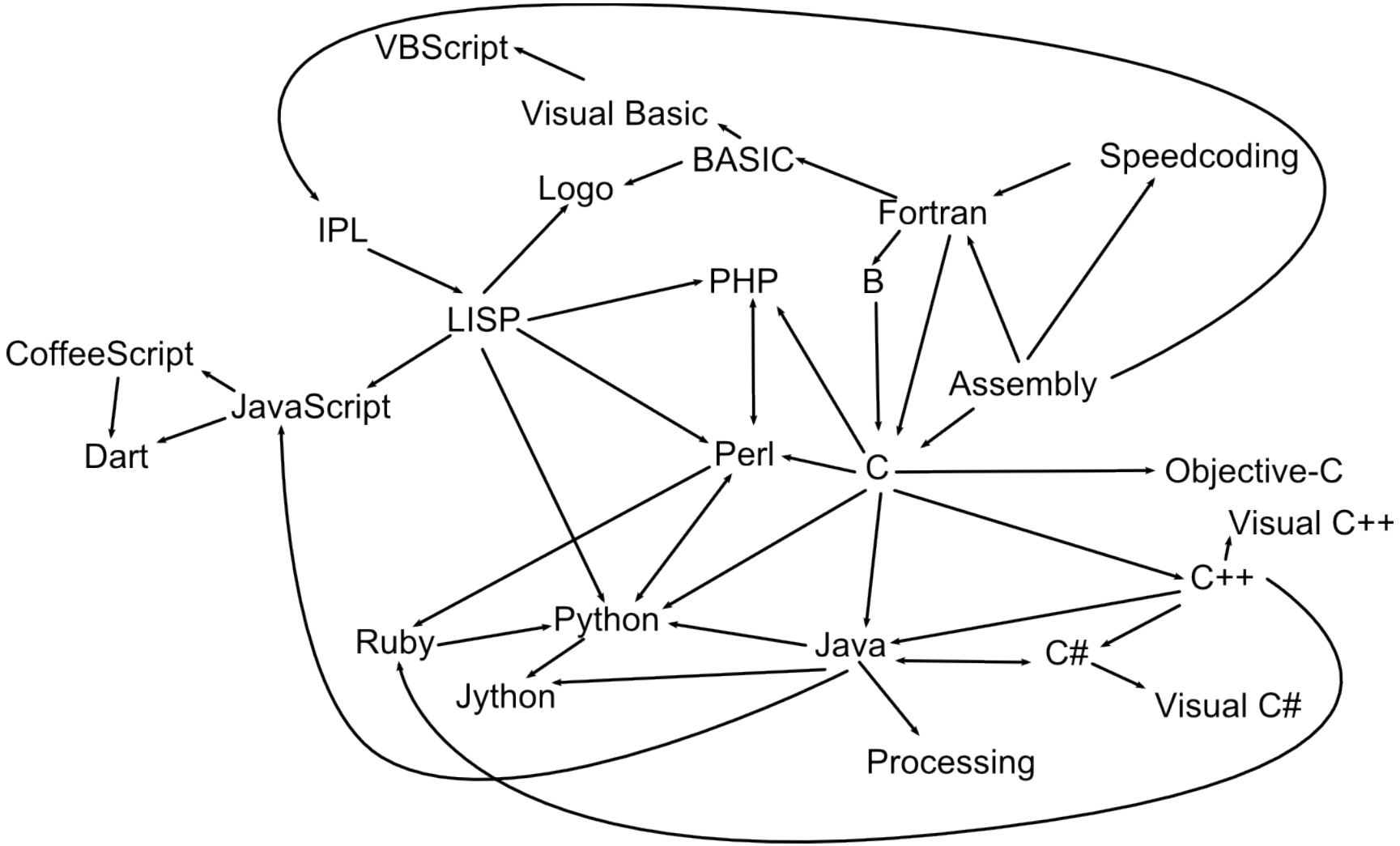
Server: *gamelab.hhs.se*
Operating system: OpenBSD (*www.openbsd.org*)

Contact me (*kristaps@kth.se*) for access or use the temporary guest account.

C: Why?

1. supported by default on most (all?) UNIX systems
2. extremely good support for performance ("up", distributed computing; "down", mixing assembly)
3. many open and closed-source domain-specific supporting libraries for
4. bindings for most other modern programming languages
5. familiar syntax (most languages are based upon C)

C: What is it?



Source: *thomasinterestingblog.wordpress.com*

C: What is it?

```
int
main(void)
{
    printf("Hello, world!\n");
    return(0);
}
```

C: How do I use it?

Many resources exist to learn C. But really only one.

Books Brian Kernighan and Dennis Ritchie, *The C Programming Language*, Prentice Hall, 1988.

Case Study

Consider a trivial primality test (trial by division). Run-time: $\mathcal{O}(nm)$ where m is the mean input value.

```
for (i = 0; i < sz; i++) {
    for (j = 2; j < p[i] - 1; j++)
        if (0 == p[i] % j)
            break;
    if (j == p[i] - 1)
        printf("%d\n", p[i]);
}
```

Case Study: Problem and Solutions

Assume the algorithm is fixed. How can we improve?

1. Faster processors.
2. More processors (problem: fairness).
3. Micro-optimising (branch prediction in the loop).
4. Vector-parallelism (loop-unrolling).

Case Study: Source

```c
#include <sys/wait.h>
#include <sys/types.h>

#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>

static void f(int, int *, int, int);

int
main(void)
{
        int sz = 0, *p = NULL, v;

        while (1 == scanf("%d", &v)) {
                p = realloc(p, (sz + 1) * sizeof(int));
                p[sz++] = v;
        }

        if (0 == fork())
                f(0, p, 0, sz / 2);
        if (0 == fork())
                f(1, p, sz / 2, sz);

        wait(NULL);
        wait(NULL);
        free(p);
        exit(EXIT_SUCCESS);
}
```

Case Study: Source

```
static void
f(int id, int *p, int start, int sz)
{
        FILE *f;
        char buf[64];
        int i, j;

        snprintf(buf, sizeof(buf), "example0.%d.dat", id);

        f = fopen(buf, "w");

        for (i = start; i < sz; i++) {
                for (j = 2; j < p[i] - 1; j++)
                        if (0 == p[i] % j)
                                break;
                if (j == p[i] - 1)
                        fprintf(f, "%d\n", p[i]);
        }

        fclose(f);
        free(p);
        exit(EXIT_FAILURE);
}
```

Case Study: Environment

Consider computing the value of function f for n numbers.

```
$ scp example0.c gamelab.hhs.se:
$ ssh guest@gamelab.hhs.se
guest@gamelab.hhs.se's password:
$ cc -Wall -o example0 example0.c
$ ./example0
$ sort -n example0.*.dat >example0.dat
$ exit
Connection to gamelab.hhs.se closed.
$ scp guest@gamelab.hhs.se:example0.dat .
```

Case Study: Benchmark

How do we begin to analyse performance?

```
$ jot -b 93563 10000 | time ./example0
    0m8.76s real
    0m15.86s user
    0m0.27s system
```

We just doubled performance!