COMS W4156 Advanced Software Engineering (ASE)

September 21, 2021

Note to Self:

Test sound before class starts: https://www.youtube.com/watch?v=i6F53B1iMrY

How Can We Communicate Better During Class? (part 1)

When I lecture in the classroom, you probably hear something like:
https://www.youtube.com/watch?v=XrbumvF-Oe4

Any ideas on how to improve this?

How Can We Communicate Better During Class? (part 2)

Piazza Live Q&A has not worked for us - it's not refreshing properly on my ipad and most of you apparently don't want to use it anyway

Speaking aloud does not work very well - when you ask a question, I typically hear something like this: https://www.youtube.com/watch?v=q_BU5hR9gXE

One idea is shared google doc, at least for brainstorming about alternative tools we can use to communicate: https://docs.google.com/document/d/17nVLZAKCRbIH4wKqmkhjCpUxJeS6Z7oAgRzlHHexa-I/edit?usp=sharing

Other ideas?

How Can We Communicate Better During Class? (part 3)

I normally include short “exercises” in many lectures, particularly during the early part of the semester, where I ask students to work together with the other students sitting near them

After team projects start, I normally devote part of some lectures to team meetings, asking teams to sit together and IAs move between their teams

Last year I used zoom breakout rooms, which worked fine for team meetings but so-so for group exercises

I don’t know how to do this now, ideas?
https://docs.google.com/document/d/17nVLZAKCRbIH4wKqmkhjCpUxJeS6Z7oAgRzlHHexa-I/edit?usp=sharing

Software Engineers always work in Teams

People who write run-once programs are programmers not software engineers

Teams always have at least three members: you, the past you, and the future you

You / Past You / Future You

Motivate...

version control - It worked last week, why isn’t it working now?

coding style - Why did I name this function ‘FixThisLater’ two years ago?

code documentation - What does ‘FixThisLater’ do?

task board - Where is the sticky note with my to-do list?

....even if your team never grows

Multi-Member Teams

There are usually more members, all of which join the team at some point (not necessarily beginning) and may leave at some point (not necessarily end)

A team of 5 software engineers often has >5 other members (some shared with other teams), such as tech lead, engineering manager, product manager, tech writer, customer support and/or devops liaison, UIUX designer and/or researcher (when applicable), possibly a separate testing team, and several layers of management above each

Some teams also have student interns or trainees

Multi-Member Engineering Teams

Mandate...

version control - My code worked last week, now it doesn’t, who changed it?

coding style - Why did Jing, who left two years ago to work for Kookle, name this function ‘FixThisLater’?

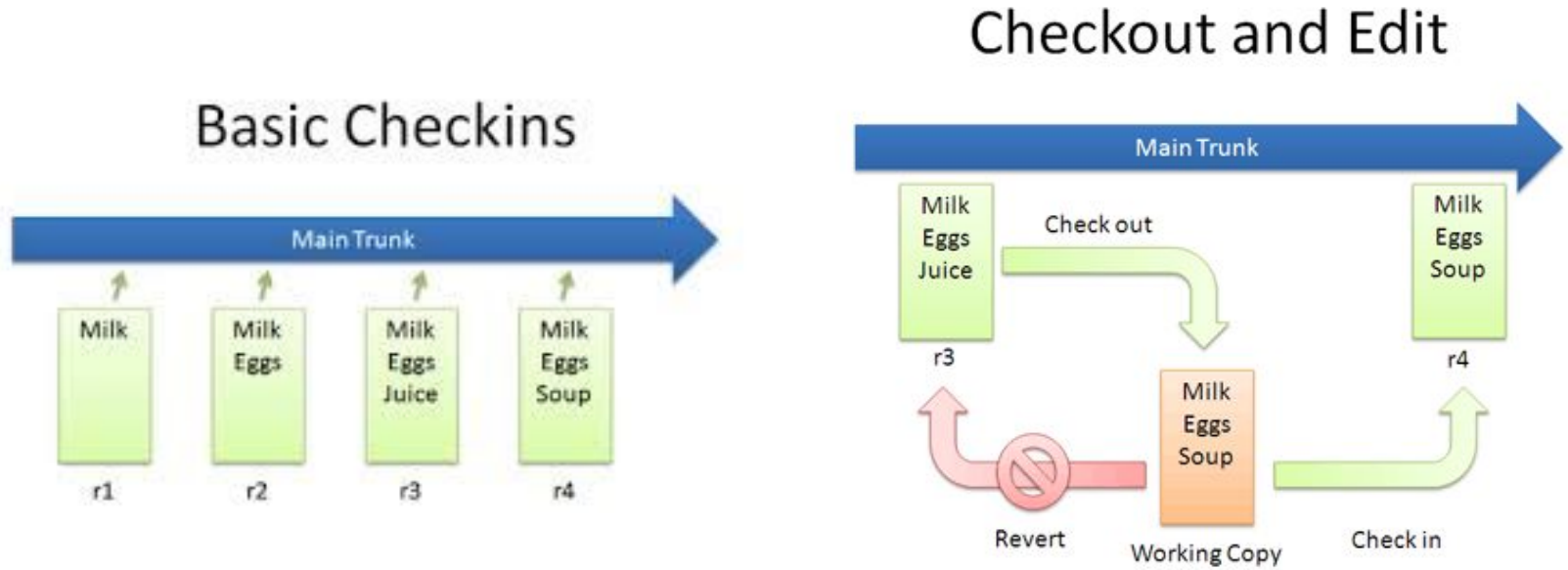
code documentation - What does ‘FixThisLater’ do?

task board - What am I supposed to be working on?

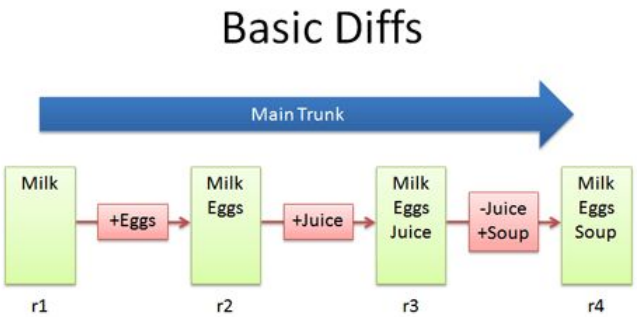
...particularly as team grows or churns

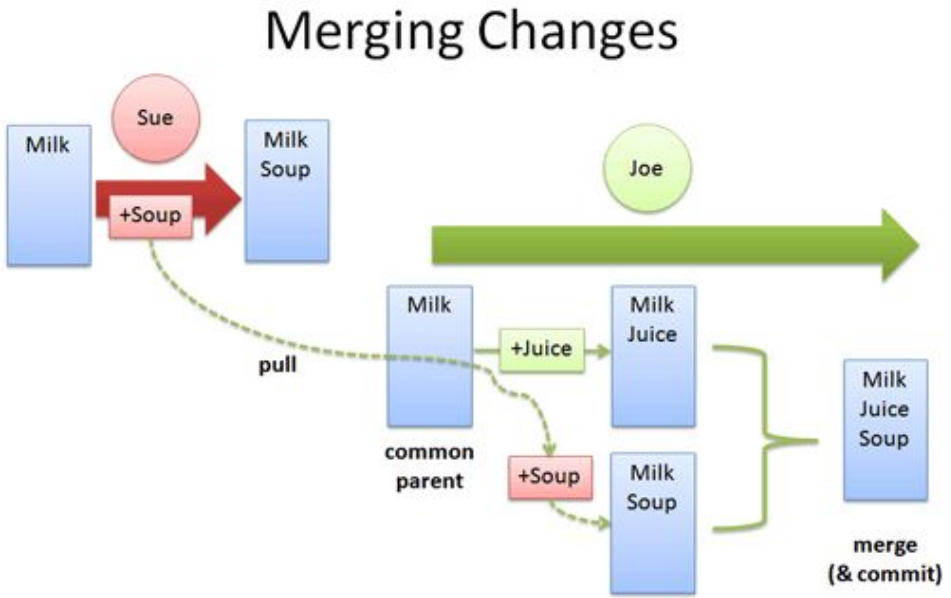
Version Control

Nearly all of you demonstrated in Homework Zero that you already understand basic version control, so just a quick refresher



Version Control





Example Auto-Merge

```
int main(){
   char* ptr;
   FILE* data;
   char buf[100];
   ….some code….
   data = fopen("data.txt", "r");
   if(fgets(buf,100,data) == NULL)
      return 1;
   printf("Read: %s", buf);
   fclose(data);
   data = fopen("data.txt", "w");
   fwrite("juice\n", 6, 1, data);
   fclose(data);
   return 0;
   }
```

```
int main(){
   char* ptr;
   FILE* data;
   char buf[100];
   ….some code...
   data = fopen("data.txt", "r");
   if(fgets(buf,100,data) == NULL)
      return 1;
   printf("Read: %s", buf);
   fclose(data);
   data = fopen("data.txt", "w");
   fwrite("soup\n", 7, 1, data);
   fclose(data);
   return 0;
   }
```

Example Probably Needs Manual Resolution

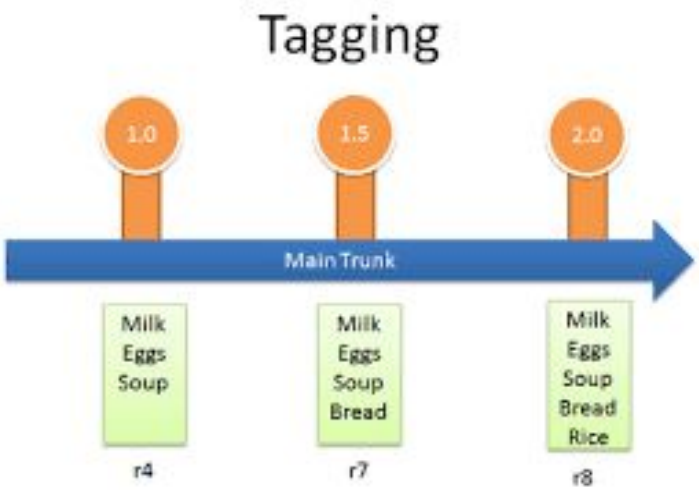
```
/* original */
void display(char* str){
   write(1, str, strlen(str));
}
int main(){
   char* msg_ok = "Ok!\n";
   char* user = calloc(256, 1);
   char* pass = calloc(256, 1);
   strncpy(user, "default", 8);
   strncpy(pass, "default", 8);
   ...some code...
   write(2, msg_ok, strlen(msg_ok));
   display(user);
   return 0;
}
```

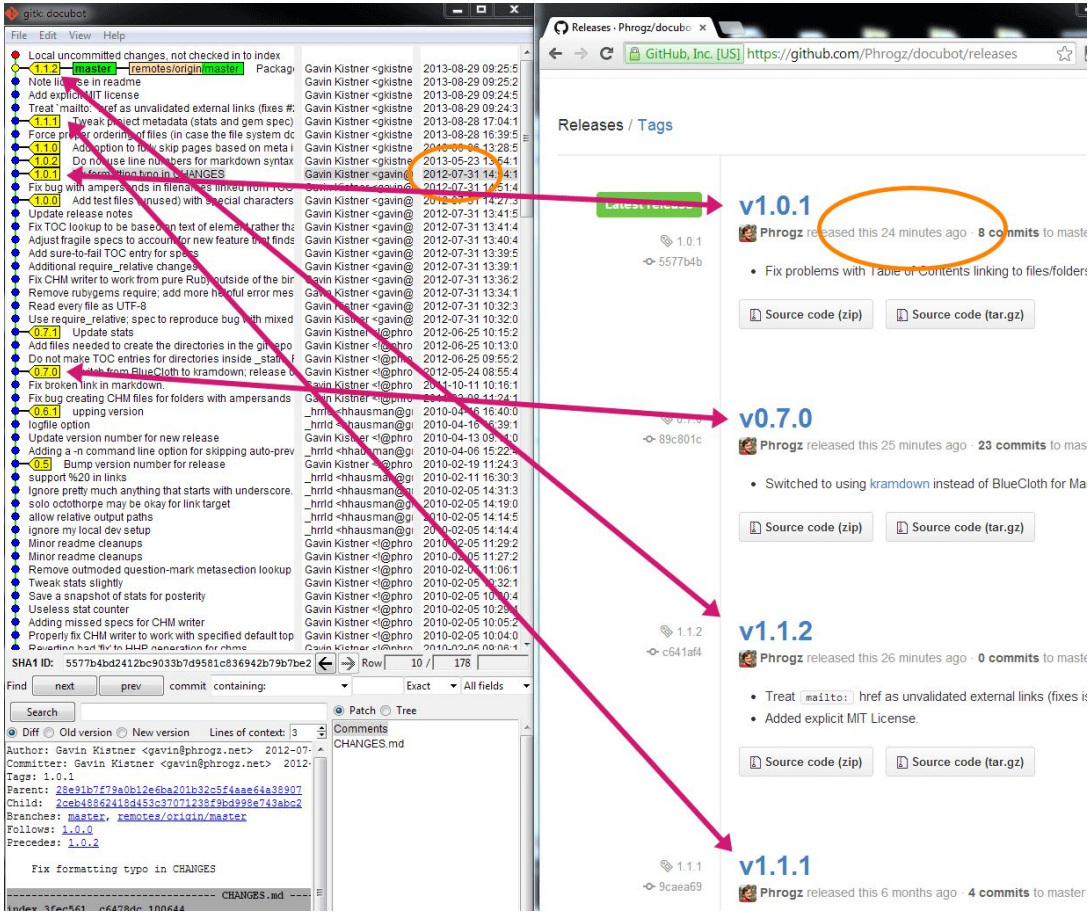
```
/* concurrent change */
void display(char* str){
   write(1, str, strlen(str));
}
int main(){
   char* msg_ok = "Ok!\n";
   char* user = calloc(256, 1);
   char* pass = calloc(256, 1);
   strncpy(user, "default", 8);
   strncpy(pass, "default", 8);
   …. some code...
   read(0, user, 255);
   read(0, pass, 255);
   write(2, msg_ok, strlen(msg_ok));
   display(user);
   return 0;
}
```

```
/* concurrent change */
void display(char* str){
   write(1, str, strlen(str));
}
int main(){
   char* msg_ok = "Ok!\n";
   char* user = calloc(256, 1);
   char* passwd = calloc(256, 1);
   strncpy(user, "default", 8);
   strncpy(passwd, "default", 8);
   ...some code...
   read(0, passwd, 255);
   write(2, msg_ok, strlen(msg_ok));
   display(user);
   return 0;
}
```

Release Tags

When you submit an iteration or present a demo, **tag** the revisions that contributed to that milestone so you can return to it later



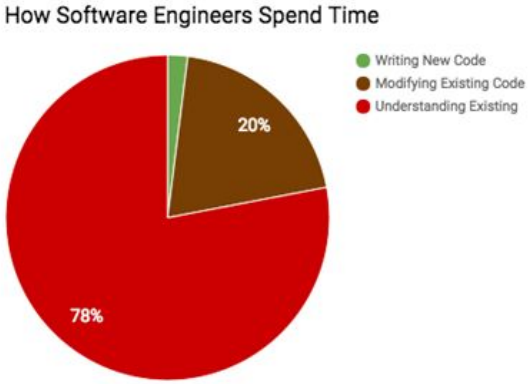


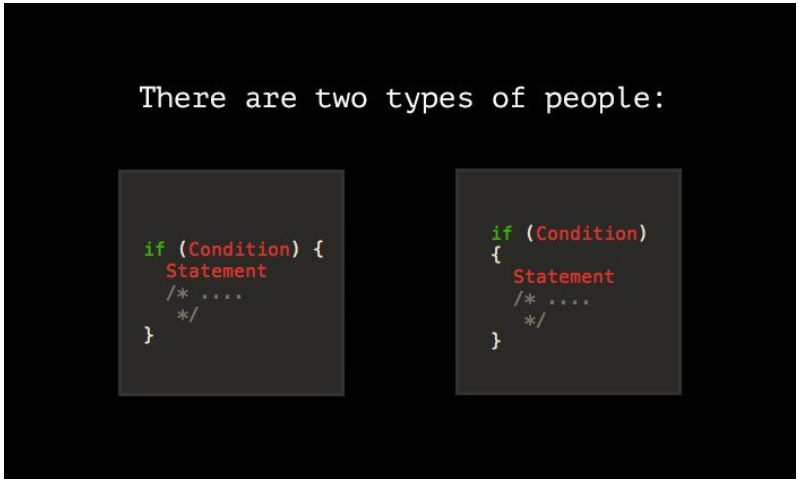
Coding Style

Governs how and when to use comments and whitespace (indentation, blank lines), proper naming of variables and functions (e.g., camelCase, snake_case), code grouping and organization, file/folder structure

Compliance with a team's coding style helps other developers understand your code and vice versa, even helps your future self understand your past self's code

See Google's [many style guides]





Your Eyes Can Deceive You

Which tests pass - neither, first, second, both?

```
public class TestBadCode {

    public int calculateFoo(int x, int y, boolean increment){
        if(increment)
            x++;
            x *=2;
        x += y;
        return x;
    }

    @Test
    public void test() {
        assertEquals(calculateFoo(3, 5, true), 13);
        assertEquals(calculateFoo(3, 5, false), 8);
    }

}
```

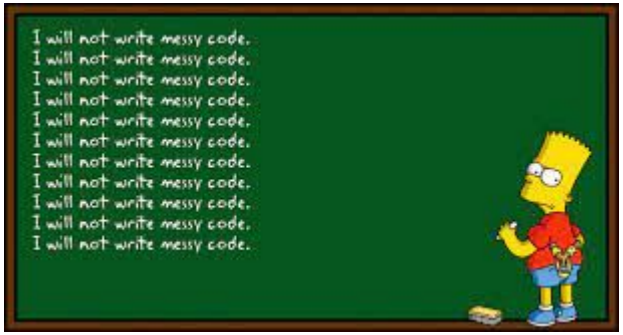
What is wrong with this code?

```
switch(value) {
    case 1:
        doSomething();

    case 2:
        doSomethingElse();
        break;

    default:
        doDefaultThing();
}
```

What Does This Code Do?

```
#include <stdio.h>

#define N(a)       "%"#a"$hhn"
#define O(a,b)     "%10$"#a"d"N(b)
#define U          "%10$.*37$d"
#define G(a)       "%"#a"$s"
#define H(a,b)     G(a)G(b)
#define T(a)       a a
#define s(a)       T(a)T(a)
#define A(a)       s(a)T(a)a
#define n(a)       A(a)a
#define D(a)       n(a)A(a)
#define C(a)       D(a)a
#define R          C(C(N(12)G(12)))
#define o(a,b,c)   C(H(a,a))D(G(a))C(H(b,b)G(b))n(G(b))O(32,c)R
#define SS         O(78,55)R "\n\033[2J\n%26$s";
#define E(a,b,c,d) H(a,b)G(c)O(253,11)R G(11)O(255,11)R H(11,d)N(d)O(253,35)R
#define S(a,b)     O(254,11)H(a,b)N(68)R G(68)O(255,68)N(12)H(12,68)G(67)N(67)

char* fmt = O(10,39)N(40)N(41)N(42)N(43)N(66)N(69)N(24)O(22,65)O(5,70)O(8,44)N(
            45)N(46)N    (47)N(48)N(    49)N( 50)N(     51)N(52)N(53    )O( 28,
            54)O(5,        55) O(2,    56)O(3,57)O(      4,58 )O(13,    73)O(4,
            71 )N(   72)O   (20,59    )N(60)N(61)N(       62)N (63)N    (64)R R
            E(1,2,   3,13   )E(4,    5,6,13)E(7,8,9        ,13)E(1,4    ,7,13)E
            (2,5,8,        13)E(    3,6,9,13)E(1,5,         9,13)E(3    ,5,7,13
            )E(14,15,    16,23)    E(17,18,19,23)E(          20, 21,    22,23)E
            (14,17,20,23)E(15,    18,21,23)E(16,19,    22     ,23)E(    14, 18,
            22,23)E(16,18,20,    23)R U O(255 ,38)R    G (     38)O(    255,36)
            R H(13,23)O(255,    11)R H(11,36) O(254    ,36)     R G(    36 ) O(
            255,36)R S(1,14    )S(2,15)S(3, 16)S(4,    17 )S     (5,    18)S(6,
            19)S(7,20)S(8,    21)S(9    ,22)H(13,23    )H(36,    67    )N(11)R
            G(11)""O(255,    25 )R        s(C(G(11)    ))n (G(          11) )G(
            11)N(54)R C(    "aa")   s(A(   G(25)))T    (G(25))N         (69)R o
            (14,1,26)o(    15, 2,   27)o   (16,3,28    )o( 17,4,        29)o(18
            ,5,30)o(19    ,6,31)o(        20,7,32)o    (21,8,33)o       (22 ,9,
            34)n(C(U)    )N( 68)R H(    36,13)G(23)    N(11)R C(D(      G(11)))
            D(G(11))G(68)N(68)R G(68)O(49,35)R H(13,23)G(67)N(11)R C(H(11,11)G(
            11))A(G(11))C(H(36,36)G(36))s(G(36))O(32,58)R C(D(G(36)))A(G(36))SS

#define arg d+6,d+8,d+10,d+12,d+14,d+16,d+18,d+20,d+22,0,d+46,d+52,d+48,d+24,d\
            +26,d+28,d+30,d+32,d+34,d+36,d+38,d+40,d+50,(scanf(d+126,d+4),d+(6\
            -2)+18*(1-d[2]%2)+d[4]*2),d,d+66,d+68,d+70, d+78,d+80,d+82,d+90,d+\
            92,d+94,d+97,d+54,d[2],d+2,d+71,d+77,d+83,d+89,d+95,d+72,d+73,d+74\
            ,d+75,d+76,d+84,d+85,d+86,d+87,d+88,d+100,d+101,d+96,d+102,d+99,d+\
            67,d+69,d+79,d+81,d+91,d+93,d+98,d+103,d+58,d+60,d+98,d+126,d+127,\
            d+128,d+129

char d[538] = {1,0,10,0,10};

int main() {
    while(*d) printf(fmt, arg);
}
```

Style Checkers



Most languages have automated tools that detect deviations from a prescribed coding style, e.g., checkstyle

The auto-merge feature of git and other version control systems depend on style compliance to avoid superficial, format-related merge conflicts - so should always run style checking and fix any problems before commiting code changes

Running the style checker (and doing various other things) prior to each commit can be automated for github with a git pre-commit hook

Code Documentation

See Google's Documentation Best Practices and Engineering Practices
- "*Documentation is the story of your code*"

A non-trivial method (function, procedure, subroutine, etc.) has *inline* comments that explain "why" the code is written this way or "why" it works this way - intended audience is future developers who modify

Methods have a header (Javadoc, docstring, etc.) that explains "what" the method does and "how" to use it - intended audience is future developers who use or modify

Every behavior documented in a method header has corresponding test cases to verify that behavior, and the test cases themselves are documented with what arguments the method takes, what it returns, any "gotchas" or restrictions, and what exceptions it can throw or errors it can return

Code Documentation

Classes (modules, files, etc.) document the class as a whole, overview of what the class does and examples of how it might be used (both simple and advanced, if applicable) - intended audience is future developers who use or modify

Folders (directories) have a README.md file that states What is this directory intended to hold? Which files should the developer look at first? Are some files an API? Who maintains this directory and where I can learn more?

Task Boards

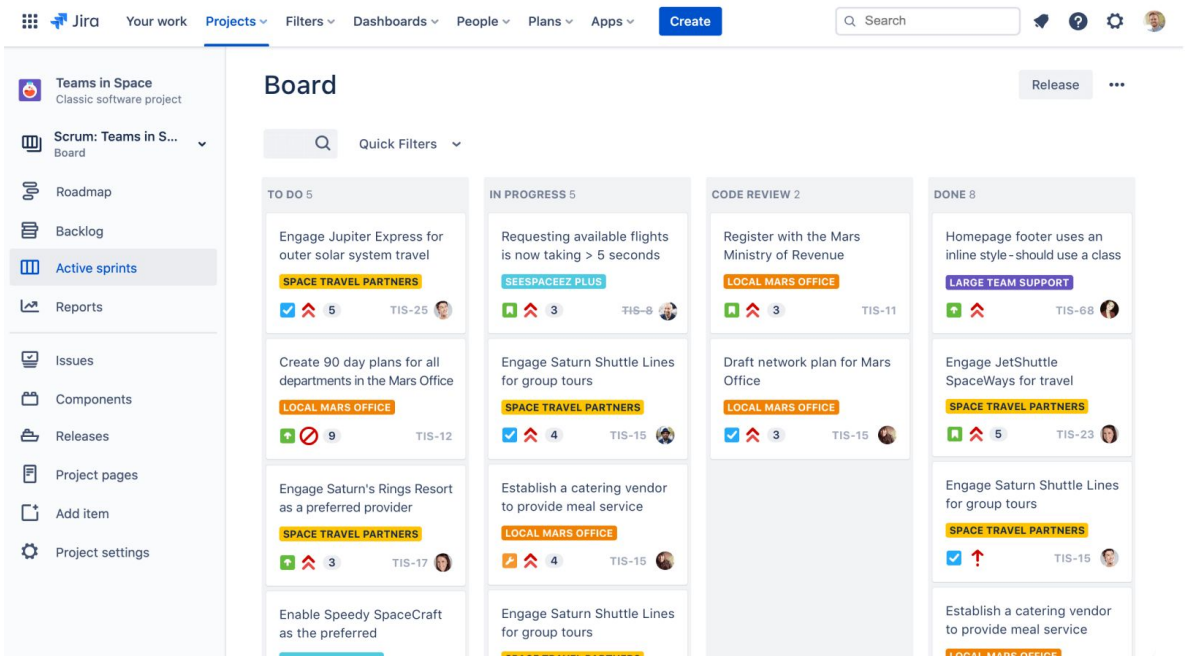
“Issue Tracking” system documents who is working on what this iteration

Tasks boards show to-do, in-progress, etc. columns of tickets - each with title, priority, who assigned to, tags, description, project its part of, and links to other tickets, revisions and/or releases

Tickets typically written and assigned by team leader or product owner

JIRA and Trello probably best-known issue tracking systems, there’s free versions for small teams

Github has built-in “issues” and “projects”



You / Past You / Future You

Motivate...

version control - It worked last week, why isn’t it working now?

coding style - Why did I name this function ‘FixThisLater’ two years ago?

code documentation - What does ‘FixThisLater’ do?

task board - Where is the sticky note with my to-do list?

....even if your team never grows

Multi-Member Engineering Teams

Mandate...

version control - My code worked last week, now it doesn't, who changed it?

coding style - Why did Jing, who left two years ago to work for Kookle, name this function 'FixThisLater'?

code documentation - What does 'FixThisLater' do?

task board - What am I supposed to be working on?

...particularly as team grows or churns

Multi-Member Engineering Teams

Enable…

pair programming!

Ought to be called “pair software engineering”,
but “pair programming” sounds better

Used for many software engineering activities, not just programming

Some companies use pair-programming interviews between job candidate and evaluator (current employee) even if their employees do not pair-program routinely

Ideal Pair Programming

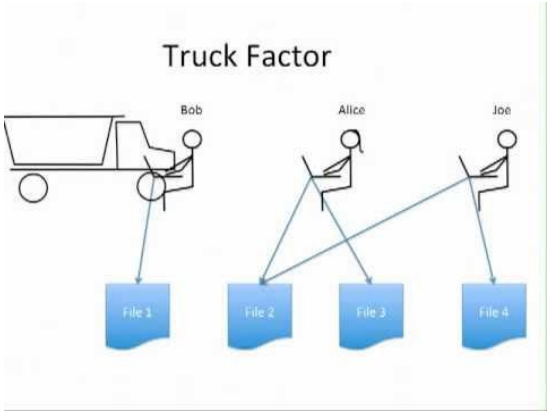
Two people sit side by side at same computer or use remote desktop sharing

- Take turns “driving” (typing) vs. “navigating” (continuous review of code or whatever working on)
- Switch roles at frequent intervals (eg, 25 minutes)
- The pair does **not** divide up the work, they do everything **together**
- If necessary to work separately on something, eg, when one partner is absent, they review together

Benefits of Pair Programming

Reduce risk through collective code ownership and maximize “truck factor” (or “bus factor”) = number of engineers that would have to disappear before project would be in serious jeopardy

- Shared responsibility to complete tasks on time
- Stay focused and on task
- Less likely to read email, surf web, etc.
- Less likely to be interrupted
- Partners expect “best practices” from each other
- Two people can solve problems that one couldn’t do alone and/or produce better solutions
- Pool knowledge resources, improve skills



Team Project

First step is Team Formation

After teams have been formed, you will propose your own project (within constraints) and proceed to develop/test/demo the project in two iterations

Individual Mini-Project

Three parts:

1. Implementing a simple game (intentionally due tomorrow, the day after program change period ends)
2. Testing the game (due September 29)
3. Saving game state (due October 6)

Note we added the requirement to submit a <= 2-minute demo video for each part (this will be used only for grading)

See Connect Four

Revisit Communication Issues

How Can We Communicate Better During Class? (part 1)

When I lecture in the classroom, you probably hear something like:
https://www.youtube.com/watch?v=XrbumvF-Oe4

Any ideas on how to improve this?

How Can We Communicate Better During Class? (part 2)

Piazza Live Q&A has not worked for us - it's not refreshing properly on my ipad and most of you apparently don't want to use it anyway

Speaking aloud does not work very well - s when you ask a question, I typically hear something like this: https://www.youtube.com/watch?v=q_BU5hR9gXE

One idea is shared google doc, at least for brainstorming about alternative tools we can use to communicate: https://docs.google.com/document/d/17nVLZAKCRbIH4wKqmkhjCpUxJeS6Z7oAgRzlHHexa-I/edit?usp=sharing

Other ideas?

How Can We Communicate Better During Class? (part 3)

I normally include short “exercises” in many lectures, particularly during the early part of the semester, where I ask students to work together with the other students sitting near them

After team projects start, I normally devote part of some lectures to team meetings, asking teams to sit together and IAs move between their teams

Last year I used zoom breakout rooms, which worked fine for team meetings but so-so for group exercises

I don’t know how to do this now, ideas?
https://docs.google.com/document/d/17nVLZAKCRbIH4wKqmkhjCpUxJeS6Z7oAgRzlHHexa-I/edit?usp=sharing