

# Tech Debt

YEGOR BUGAYENKO

Lecture #14 out of 24  
80 minutes

The slidedeck was presented by the author in this [YouTube Video](#)

All visual and text materials presented in this slidedeck are either originally made by the author or taken from public Internet sources, such as website. Copyright belongs to their respected authors.



“Shipping first time code is like going into debt. A little debt speeds development so long as it is paid back promptly with a rewrite. The danger occurs when the debt is not repaid. Every minute spent on not-quite-right code counts as interest on that debt.”

— Ward Cunningham, *Experience Report — The WyCash portfolio management system*, OOPSLA, 1992

## Puzzle Driven Development: Motivating Example

### Commit #1:

```

1 int fibonacci(int n) {
2     if (n <= 2) {
3         return 1;
4     }
5     // @todo I don't know
6     // what to do when "n"
7     // is larger than "2".
8     // Implement it and uncomment
9     // the assertion below.
10    return 0;
11 }
12 assert fibonacci(0) == 1;
13 assert fibonacci(2) == 1;
14 // assert fibonacci(9) == 34;

```

### Commit #2:

```

1 int fibonacci(int n) {
2     if (n <= 2) {
3         return 1;
4     }
5     if (n == 9) {
6         return 34;
7     }
8     // @todo Implement others
9     // too, but I don't know
10    // how to do it right.
11    return 0;
12 }
13 assert fibonacci(2) == 1;
14 assert fibonacci(9) == 34;
15 // assert fibonacci(10) == 55;

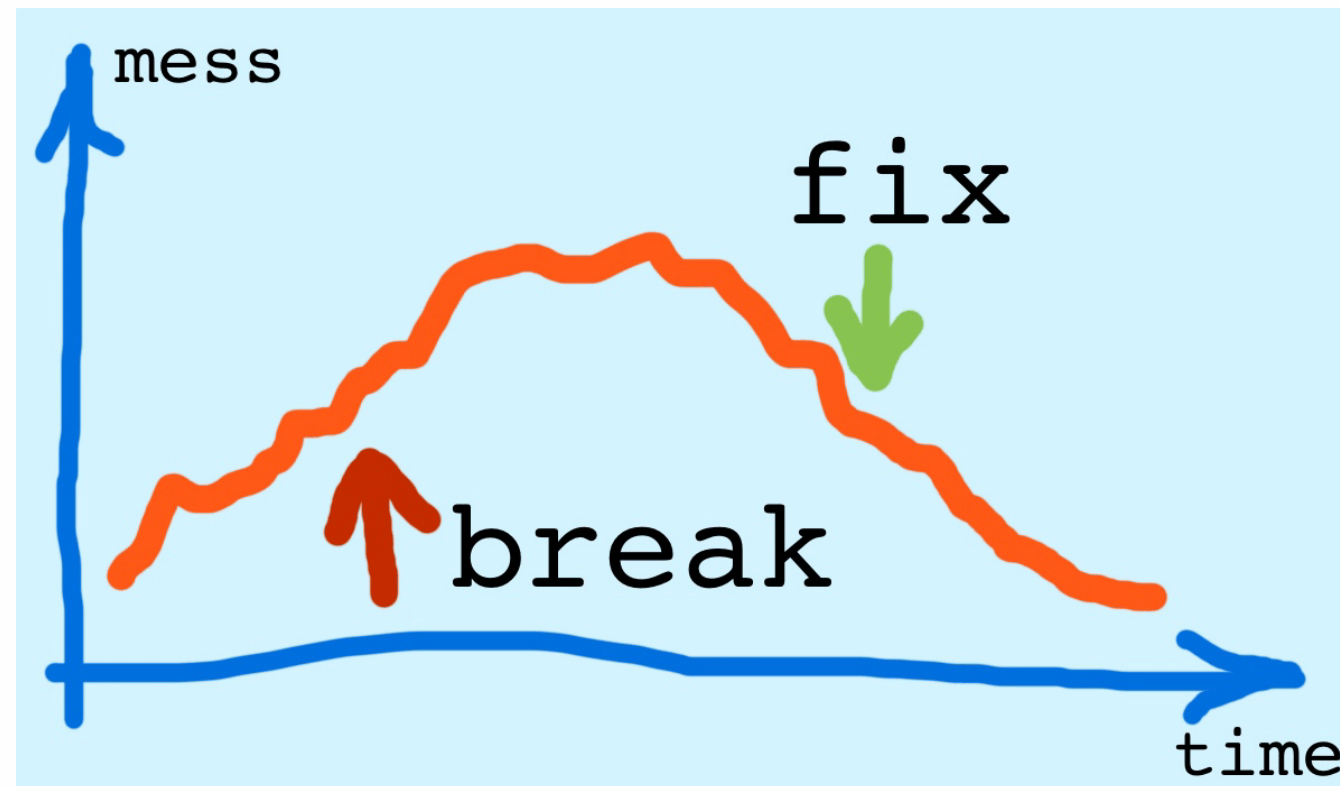
```

```

16 \end{ffcode*}
17 }
18 \par\columnbreak\par
19 Commit \#3:\par
20 {\scriptsize\begin{ffcode}
21 int fibonacci(int n) {
22     if (n <= 2) {
23         return 1;
24     }
25     return fibonacci(n-1)
26         + fibonacci(n-2);
27 }
28 assert fibonacci(0) == 1;
29 assert fibonacci(2) == 1;
30 assert fibonacci(9) == 34;
31 assert fibonacci(10) == 55;

```

## Break-and-Fix Cycle



Source: <https://www.yegor256.com/2014/04/12/puzzle-driven-development-by-roles.html>

www.Opdd.com



## PDD Pipeline

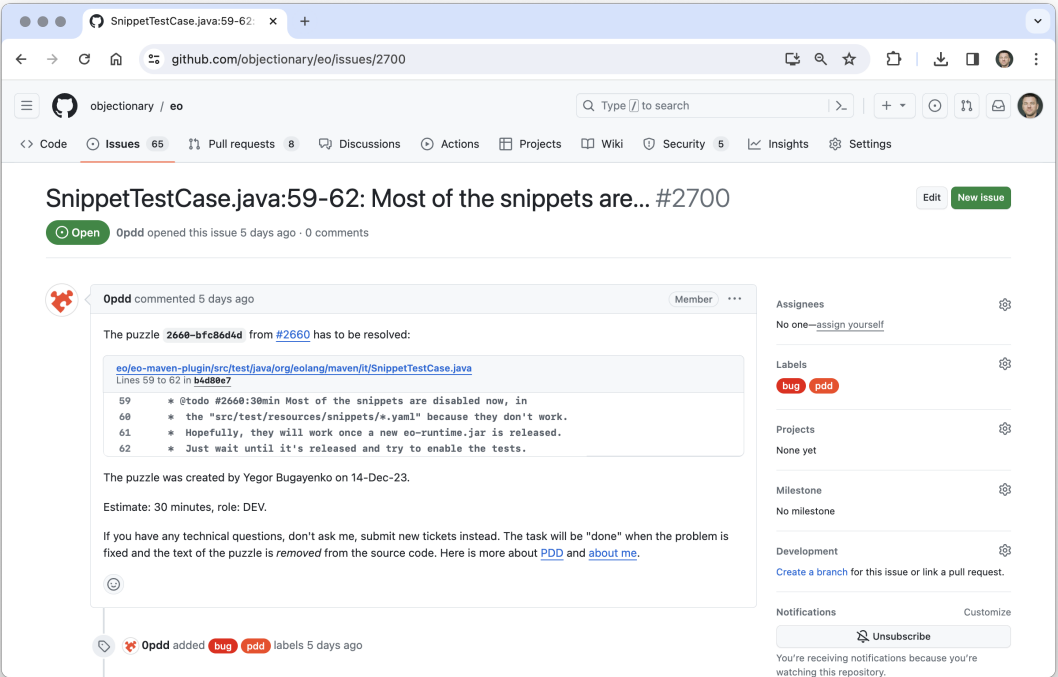


# 250+ Puzzles in objectionary/eo



Source: <https://www.0pdd.com/p?name=objectionary/eo>

# Sample Ticket Submitted by 0pdd.com to objectionary/eo



Source: <https://github.com/objectionary/eo/issues/2700>





“This paper presents the benefits of considering the entire backlog when prioritizing tasks. We employ an iterative approach using Particle Swarm Optimization to optimize a linear model with various preprocessing methods to determine the optimal model for task prioritization within a backlog.”

— Yegor Bugayenko, Mirko Farina, Artem Kruglov, Witold Pedrycz, Yaroslav Plaksin, Giancarlo Succi, *Automatically Prioritizing Tasks in Software Development*, IEEE Access, Vol. 11, 2023

## Read this:

*Automatically Prioritizing and Assigning Tasks from Code Repositories in Puzzle Driven Development*, Yegor Bugayenko, Ayomide Bakare, Arina Cheverda, Mirko Farina, Artem Kruglov, Yaroslav Plaksin, Giancarlo Succi, and Witold Pedrycz, Proceedings of the International Mining Software Repositories (MSR), 2022

*Automatically Prioritizing Tasks in Software Development*, Yegor Bugayenko, Mirko Farina, Artem Kruglov, Witold Pedrycz, Yaroslav Plaksin, and Giancarlo Succi, IEEE Access, Vol. 11, 2023

Puzzle Driven Development (2010)

PDD by Roles (2014)

PDD in Action (2017)

# References