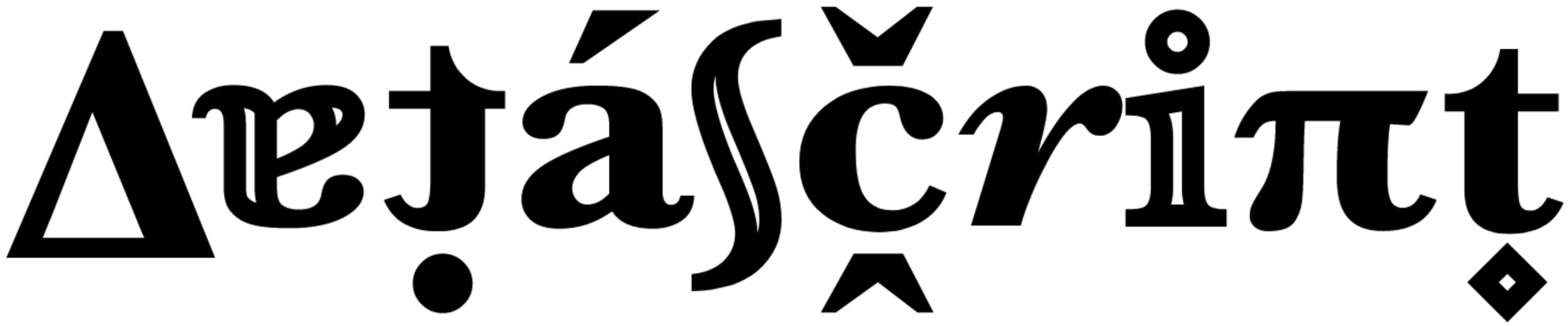
Обретение навыков

Санкт-Петербург
Май 2018
@nikitonsky

Никита Прокопов

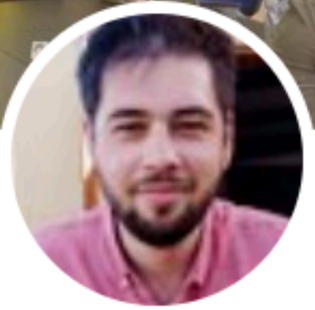

FIRA CODE
≠ ⟶» ++ :=




Δæṭáʃčriπṭ


RUM





Nikita
@nikitonsky

Am I a good person? No. But do I try to be better every day? Also no. // Clojure, DataScript, Rum, FiraCode, AnyBar,
grumpy.website

Tweets	Following	Followers
21.1K	304	4,939

@nikitonsky

ADA084551

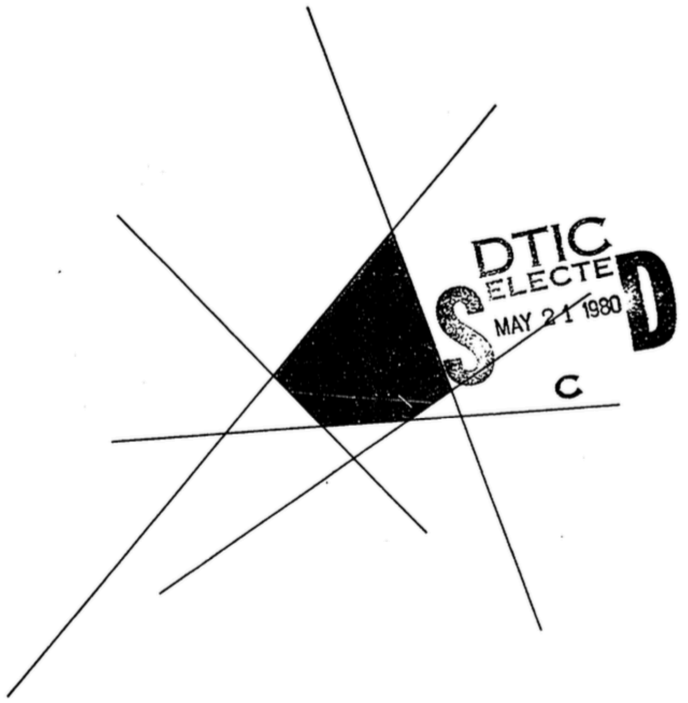
ORC 80-2
FEBRUARY 1980

LEVEL II

12

A FIVE-STAGE MODEL OF THE MENTAL ACTIVITIES INVOLVED IN DIRECTED SKILL ACQUISITION

by
STUART E. DREYFUS
and
HUBERT L. DREYFUS



DTIC
ELECTE
MAY 21 1980
S D
C

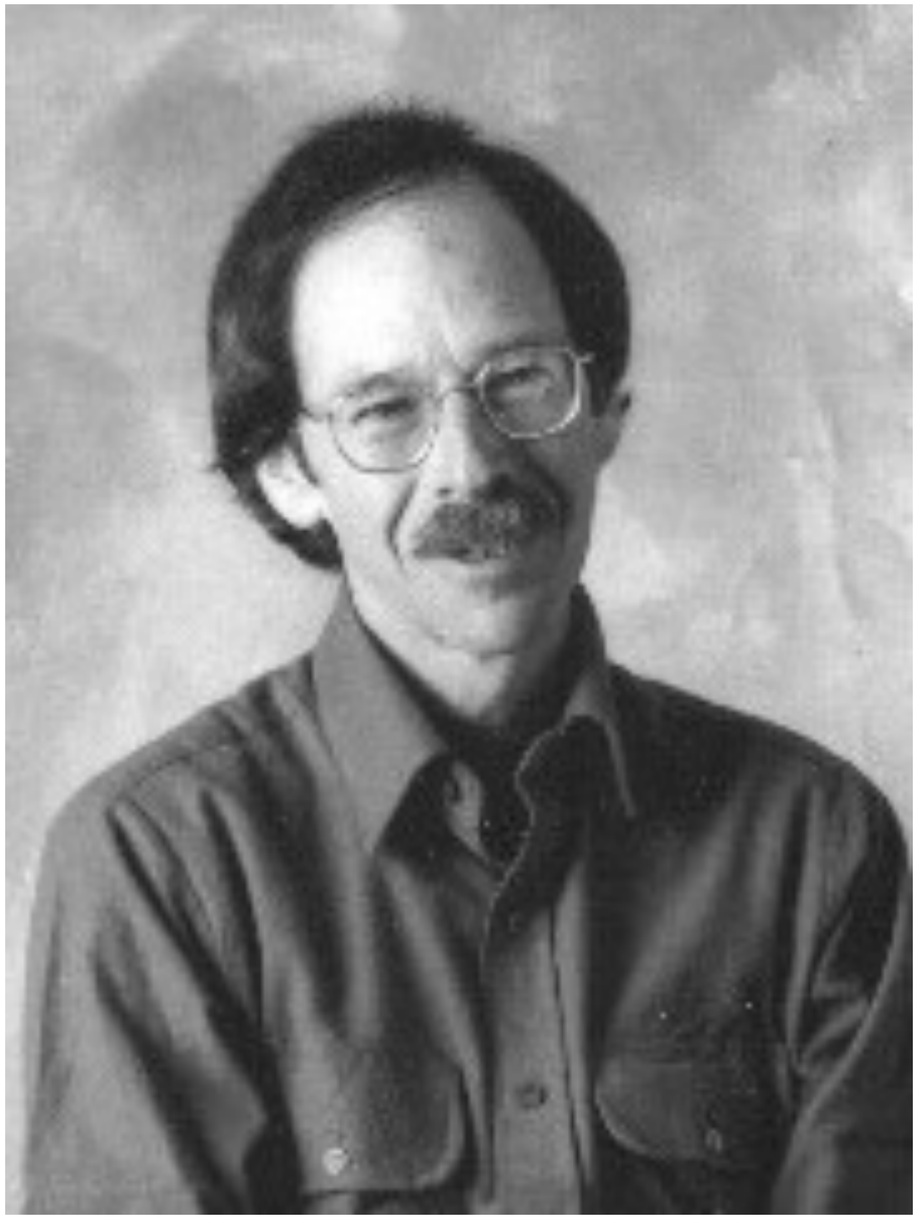
This document has been approved for public release and sale; its distribution is unlimited.

DTIC FILE COPY

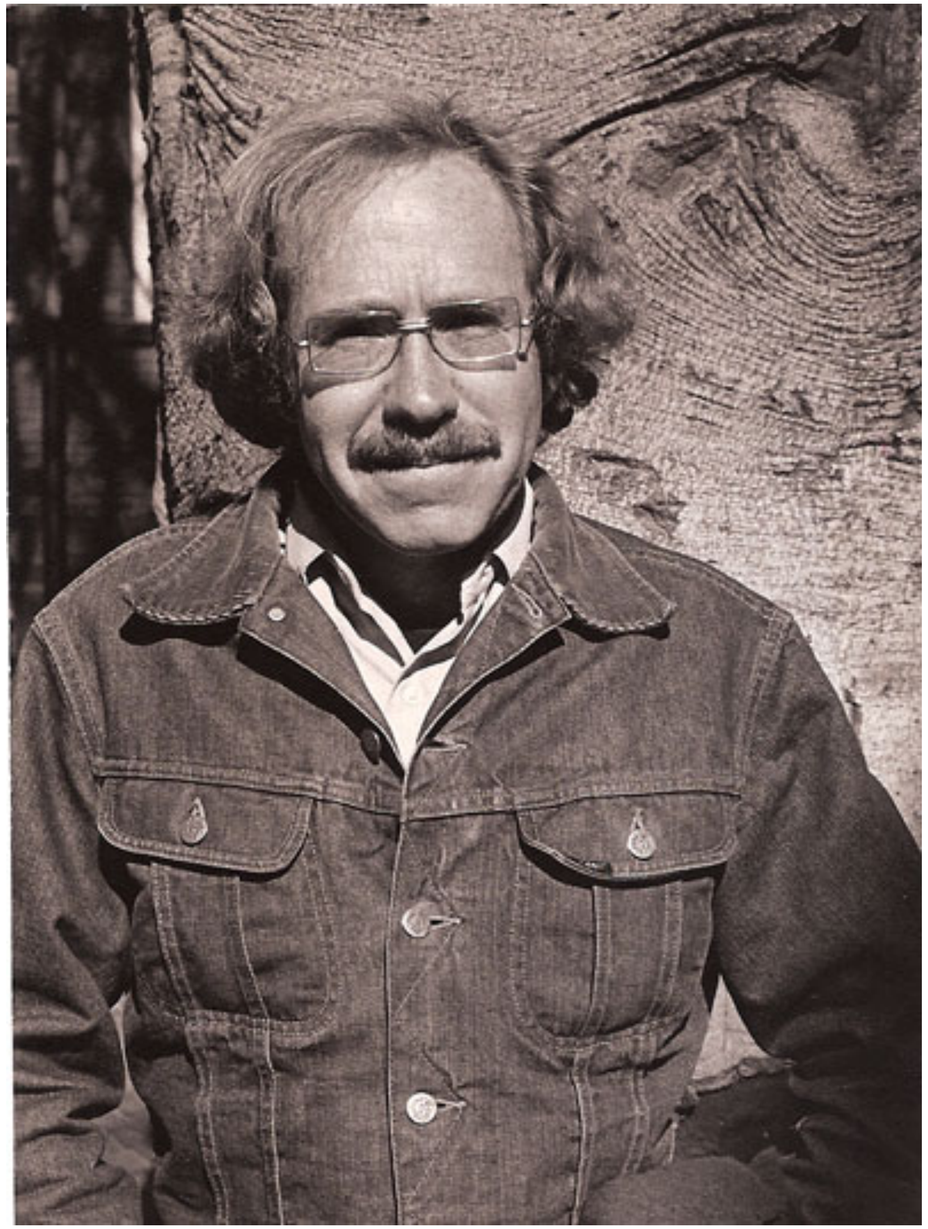
OPERATIONS RESEARCH CENTER

UNIVERSITY OF CALIFORNIA · BERKELEY

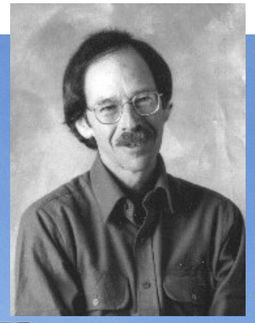
80 5 19 102

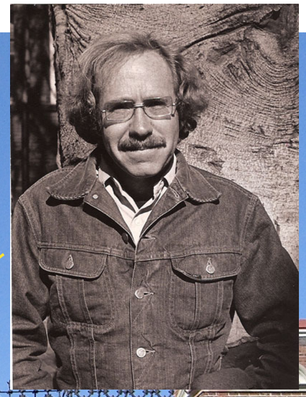


Stuart E. Dreyfus



Hubert L. Dreyfus





Эксперт

Специалист

Компетентный

Продолжающий

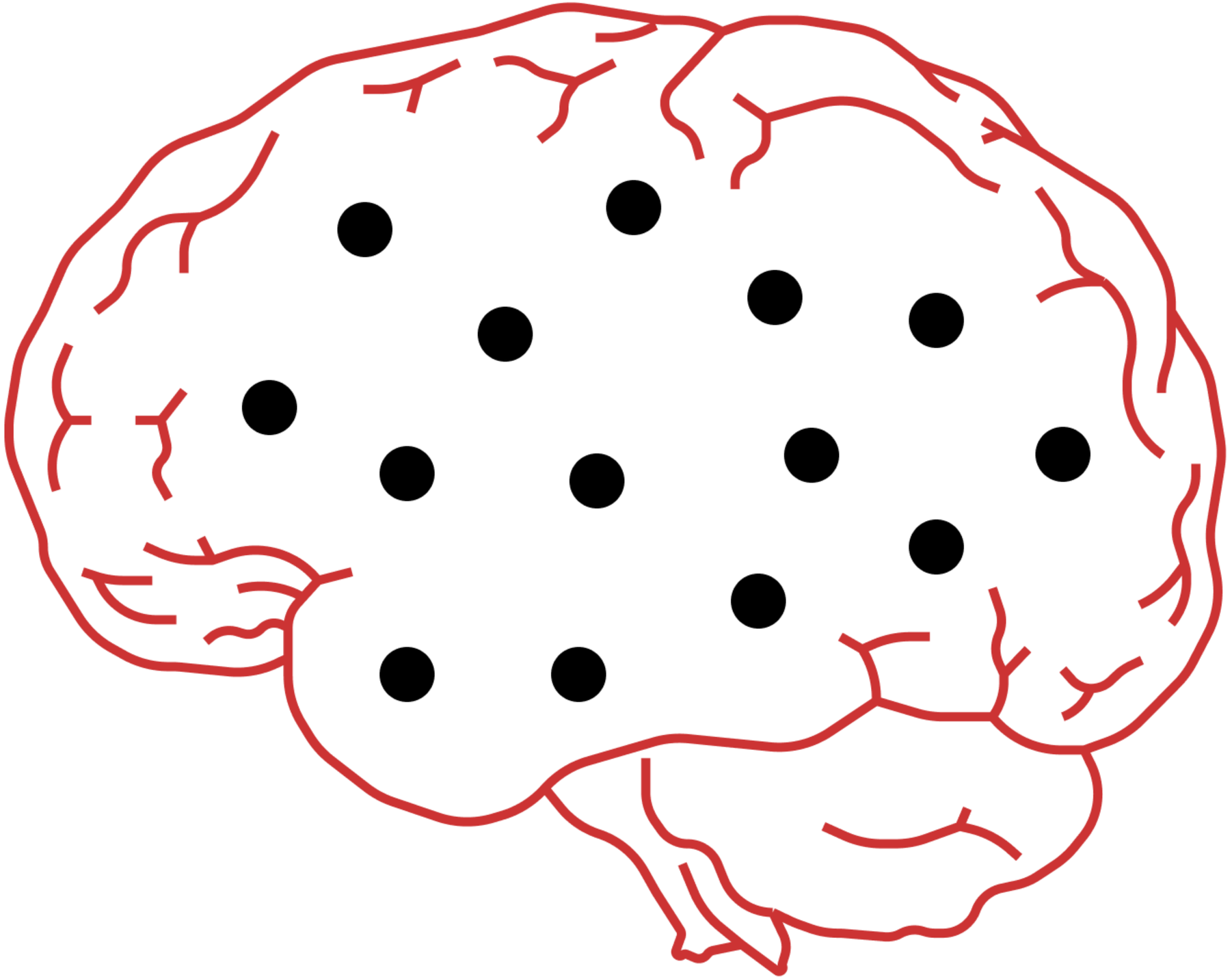
Новичок

1. Новичок



1. Новичок

- Только начинает
- Нет практического опыта
- Видит изолированные фрагменты




8
7
6
5
4
3
2
1
A
B
C
D
E
F
G
H

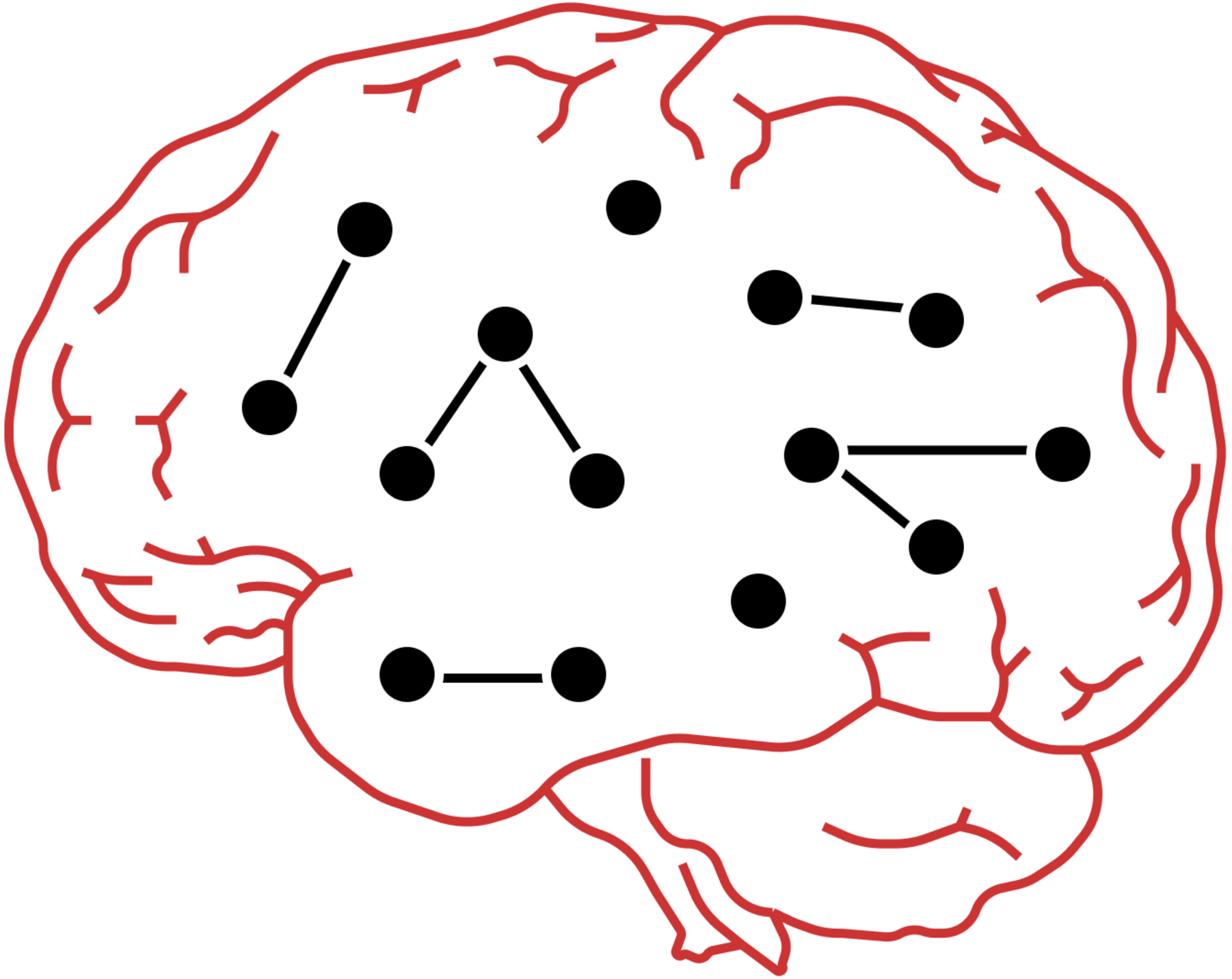
2. Продвинутый новичок

2. Продолжающий



2. Продолжающий

- Немножко опыта
- Фрагменты объединяются в аспекты
- У всех аспектов равная важность

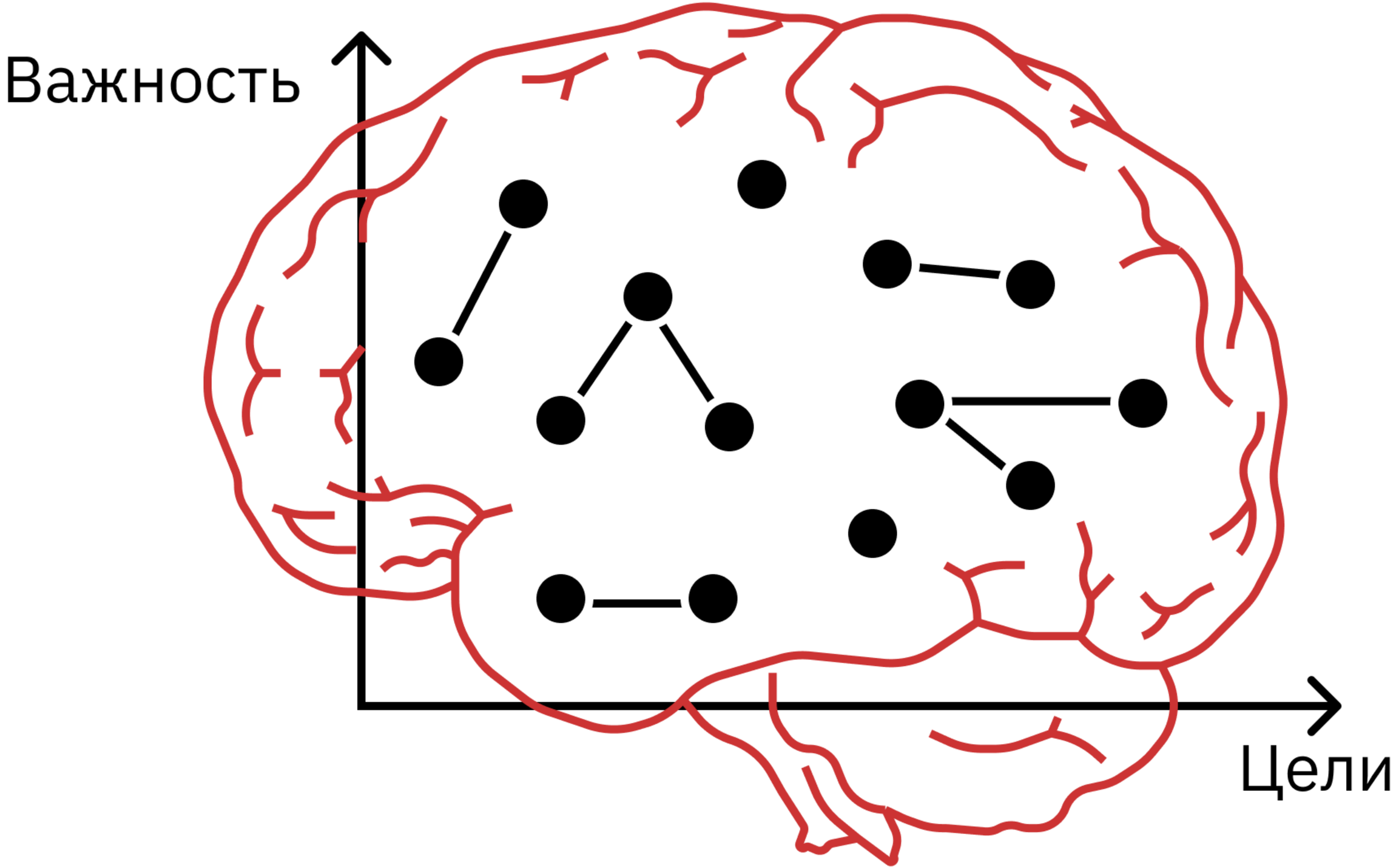



8
7
6
5
4
3
2
1
A
B
C
D
E
F
G
H



3. Компетентный

- Самостоятельная боевая единица
- Делает дело, двигается к цели
- Приоритизирует аспекты от цели
- Анализирует ситуацию, вырабатывает план
- Большинство людей здесь

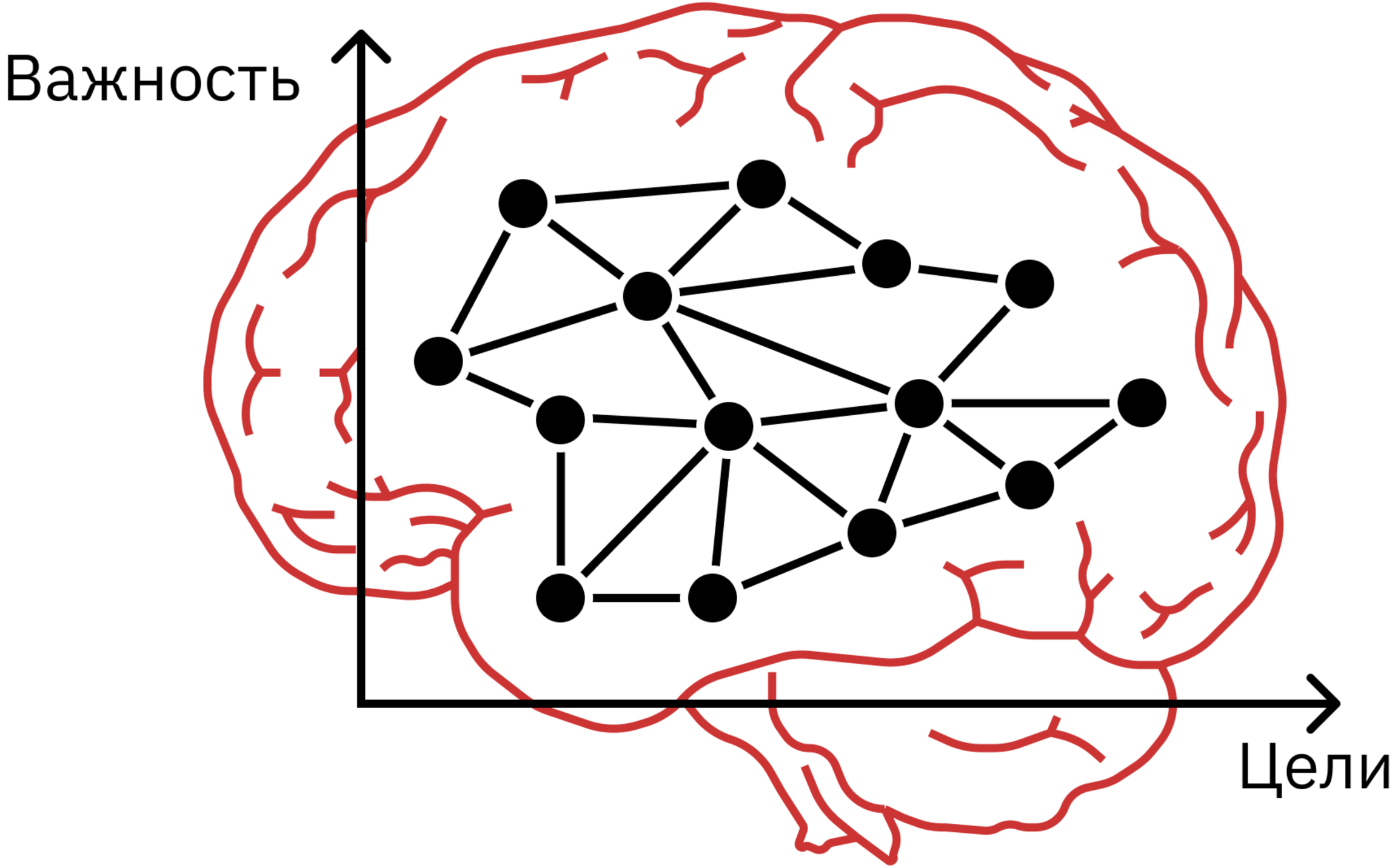

Важность
Цели


8
7
6
5
4
3
2
1
A
B
C
D
E
F
G
H



4. Специалист

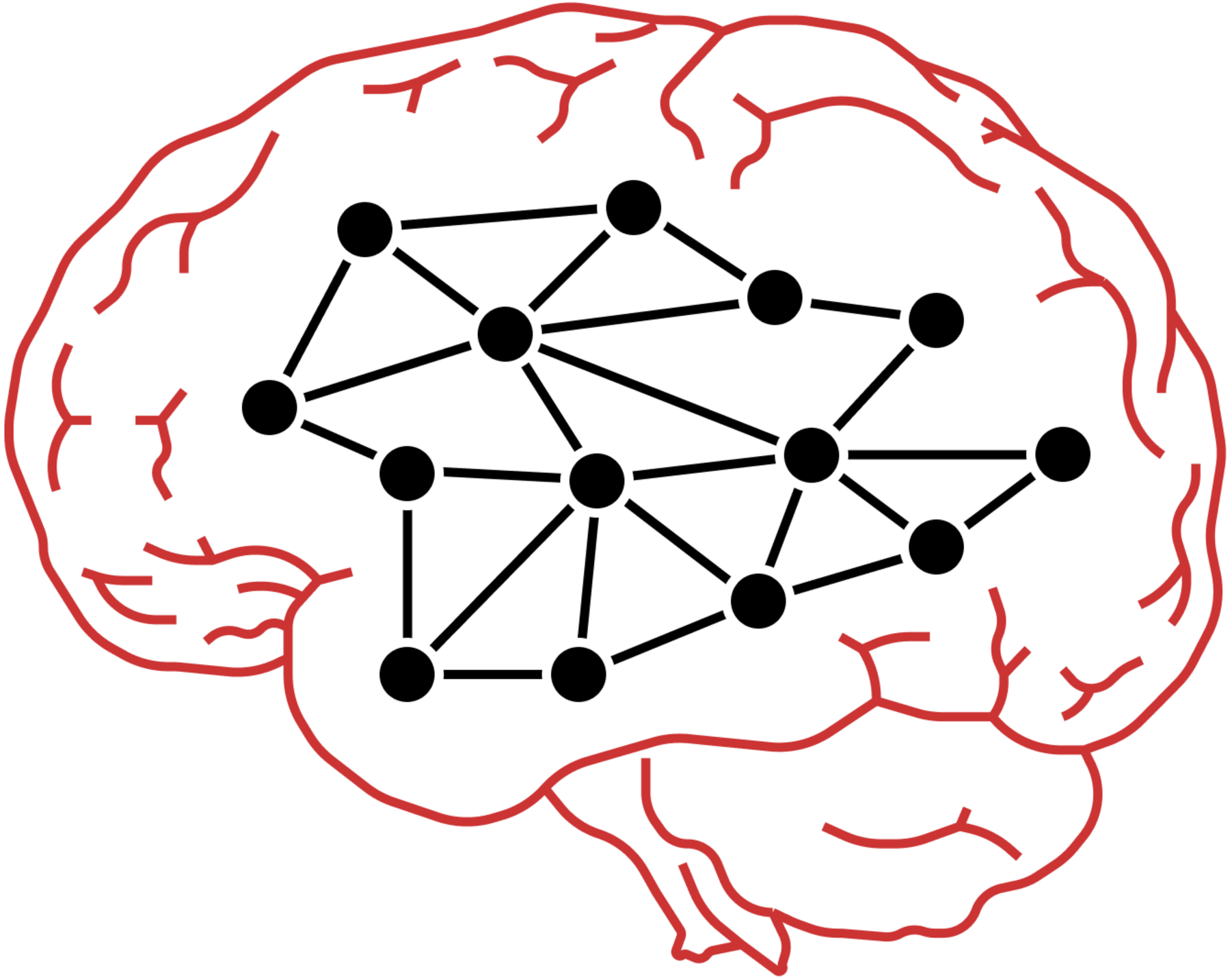
- Погрузился глубже
- Большой практический опыт
- Полагается на интуицию
- Видит «ситуацию целиком», холистически


Важность
Цели



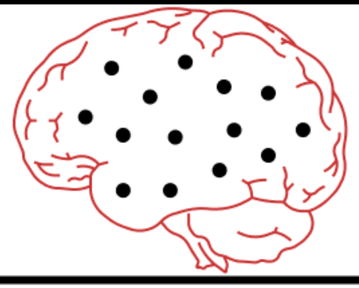
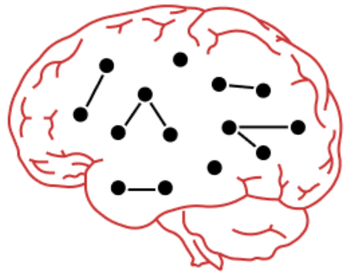
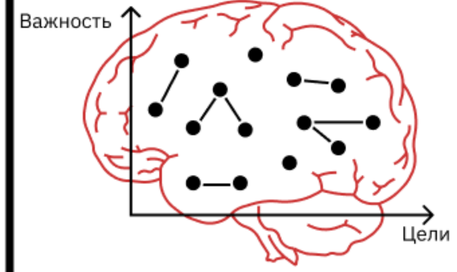
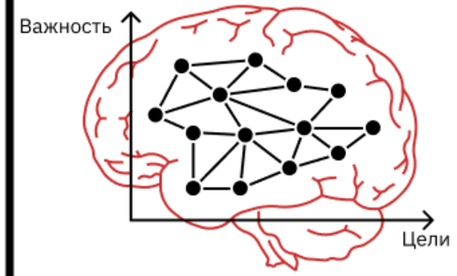
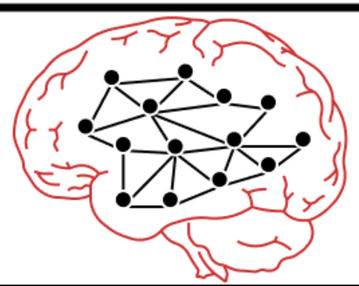
5. Эксперт

- Огромный практический опыт
- «Делается само»
- Не мыслит правилами
- Не может объяснить
- Известен в проф. кругах
- Видит «границы возможного»




8
7
6
5
4
3
2
1
A
B
C
D
E
F
G
H


ОТВ

Новичок	
Продолжающий	
Компетентный	
Специалист	
ЭКСПЕРТ	

Повторим

1. Новичок — учит правила
2. Продолжающий — пробует силы
3. Компетентный — делает дело
4. Специалист — ситуация целиком
5. Эксперт — получается само

Карьера

github.com/basecamp/handbook/blob/master/titles-for-programmers.md

1. Junior Programmer

- Thoroughly reviewed with substantial back’n’forth before merging
- **Basic language features**
- Occasional issues following patterns and approaches within existing code bases
- Tightly scoped, routine problems
- < 2 years of experience

2. Programmer

- Reviewed with the occasional need for material direction/implementation changes
- **Follows established patterns** and approaches
- Clearly defined/scoped individual features/problems
- 2-5 years of experience

3. Senior Programmer

- Work doesn't need review, general approach maybe
- **Substantial features from concept to shipping**
- Material feedback on (junior) programmers
- Deep expertise in one programming env
  + basic proficiency in at least one more
- At least 5-8 years

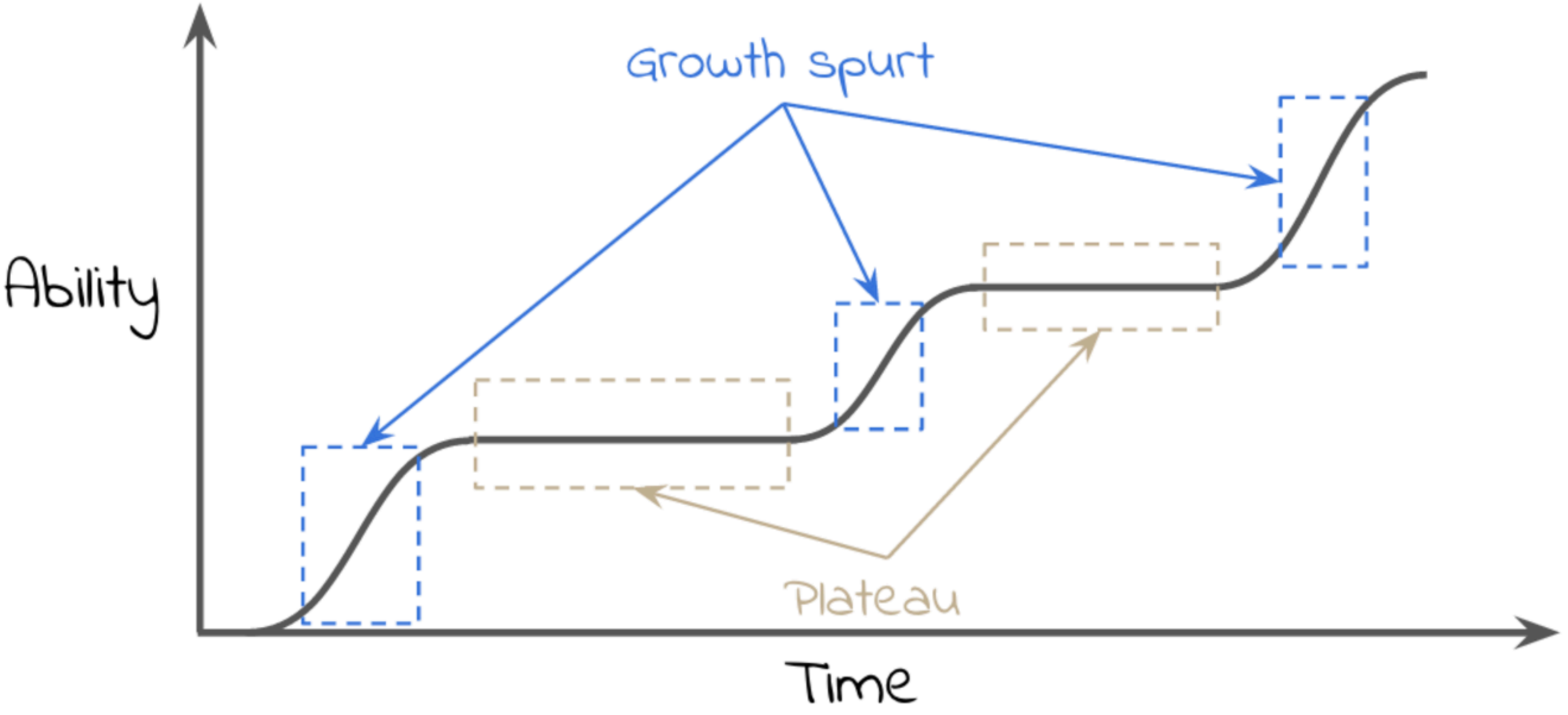
4. Lead Programmer

- Works autonomously with no need for review
- **Owning and running subsystems**
- Running small teams for substantial projects
- **Projects across multiple domains**
- **Sets professional standards** for organisation
- Deep expertise in multiple programming envs
- 8-12 years

5. Principal Programmer

- Set and direct an entire department
- Designing, owning, running new & novel systems
- Running teams for large, long-running projects
- **Recognised widely in the industry** for material contributions to the state of the art
- Invents new concepts, pushes the whole organisation forward regularly
- 12-15+ years of experience

Продвижение


Growth spurt
Ability
Plateau
Time

Новичок: следуешь правилам,
пытаешься делать хоть что-то

Продолжающий: решаешь задачи/ проблемы, тестируешь правила на прочность

Компетентный: участие в проектах

Специалист: управление проектом (контролируешь ситуацию целиком), погружение в основы



— Нужно ли программисту знать алгоритмы?



Ben Lesh 🛋👑🔥

@BenLesh



If you're a complete beginner starting off in JavaScript, there is nothing wrong with just learning a framework up front. Don't listen to people that say you have to master fundamentals first. You have to master them *eventually* but you need to have fun and be productive first.

4:17 AM - Apr 22, 2018

 3,349  1,024 people are talking about this 

Эксперт: создание проектов

и тут я проснулся...



Университет



Индустрия





Thought leader
Мемуары

Спаривание



Mr_Mig
@mr_mig_by

Follow

Джуниор: «оно работает!»

Мид: «но это же говнокод!»

Сениор: «не говнокод, а продакшен-рэди в рамках доступных ресурсов»

Тимлид: «ребята, не ссорьтесь!»

Продажи: « 🤦 »

Translate Tweet

5:05 PM - 10 May 2018

88 Retweets **325** Likes



7 88 325



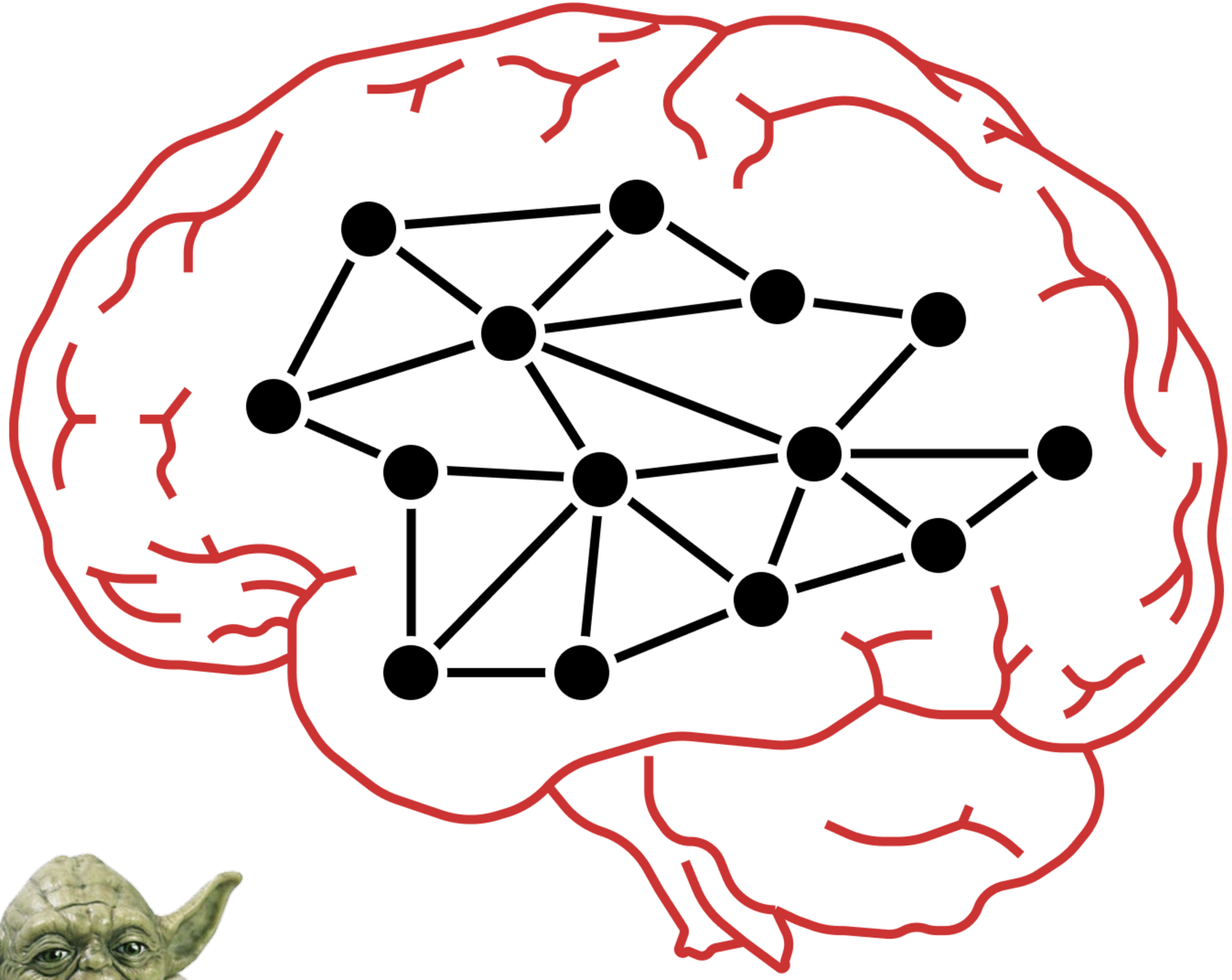
VS





VS

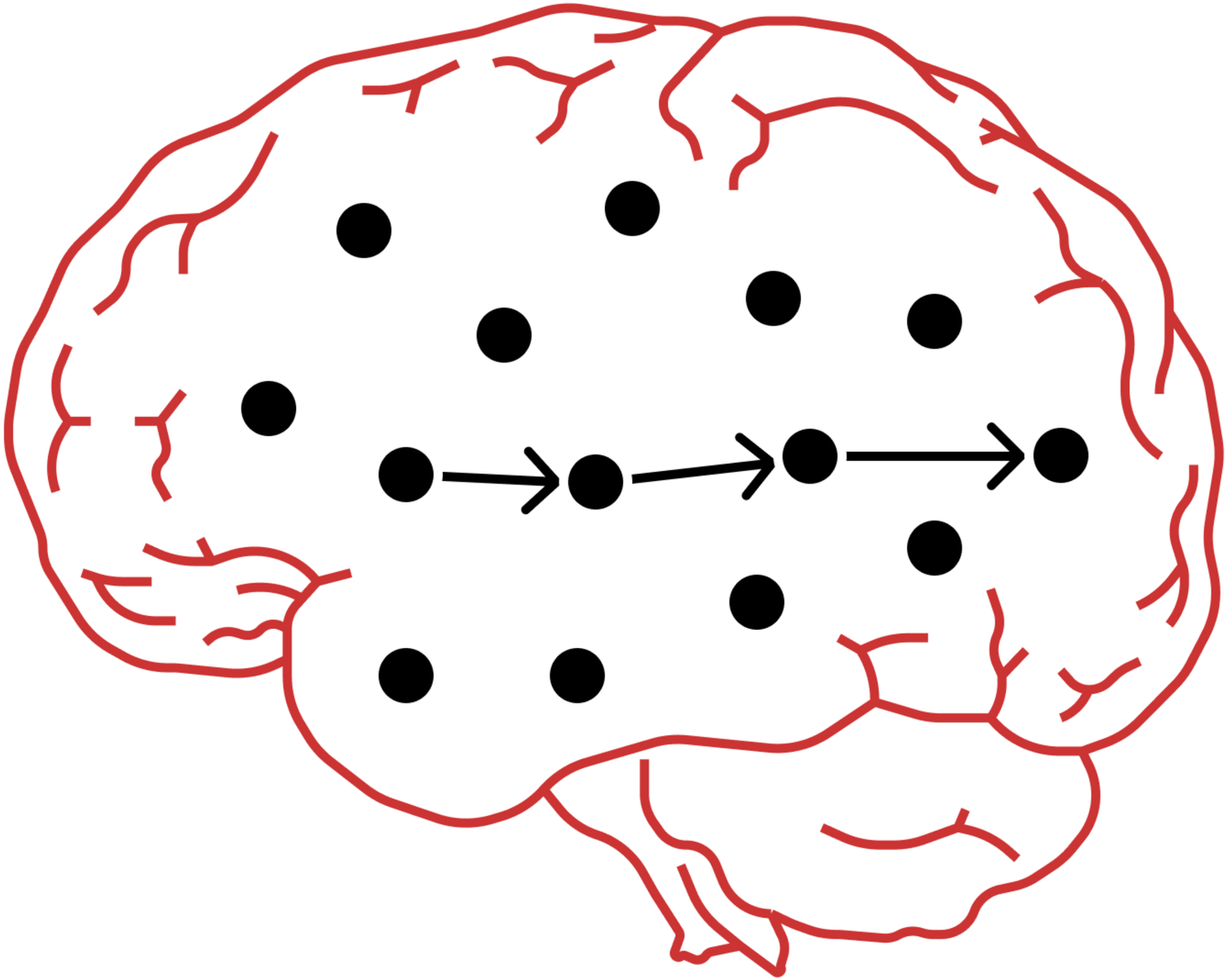




Use your fucking brain.



Handprinted on a
Korrex proof press
from original wood type
by Erik Spiekermann at
ppbk Berlin







Bob Marshall
@flowchainsensei

Mostly, I feel like the chap at the right-hand table.

11:29 AM - Sep 14, 2017

80 63 people are talking about this





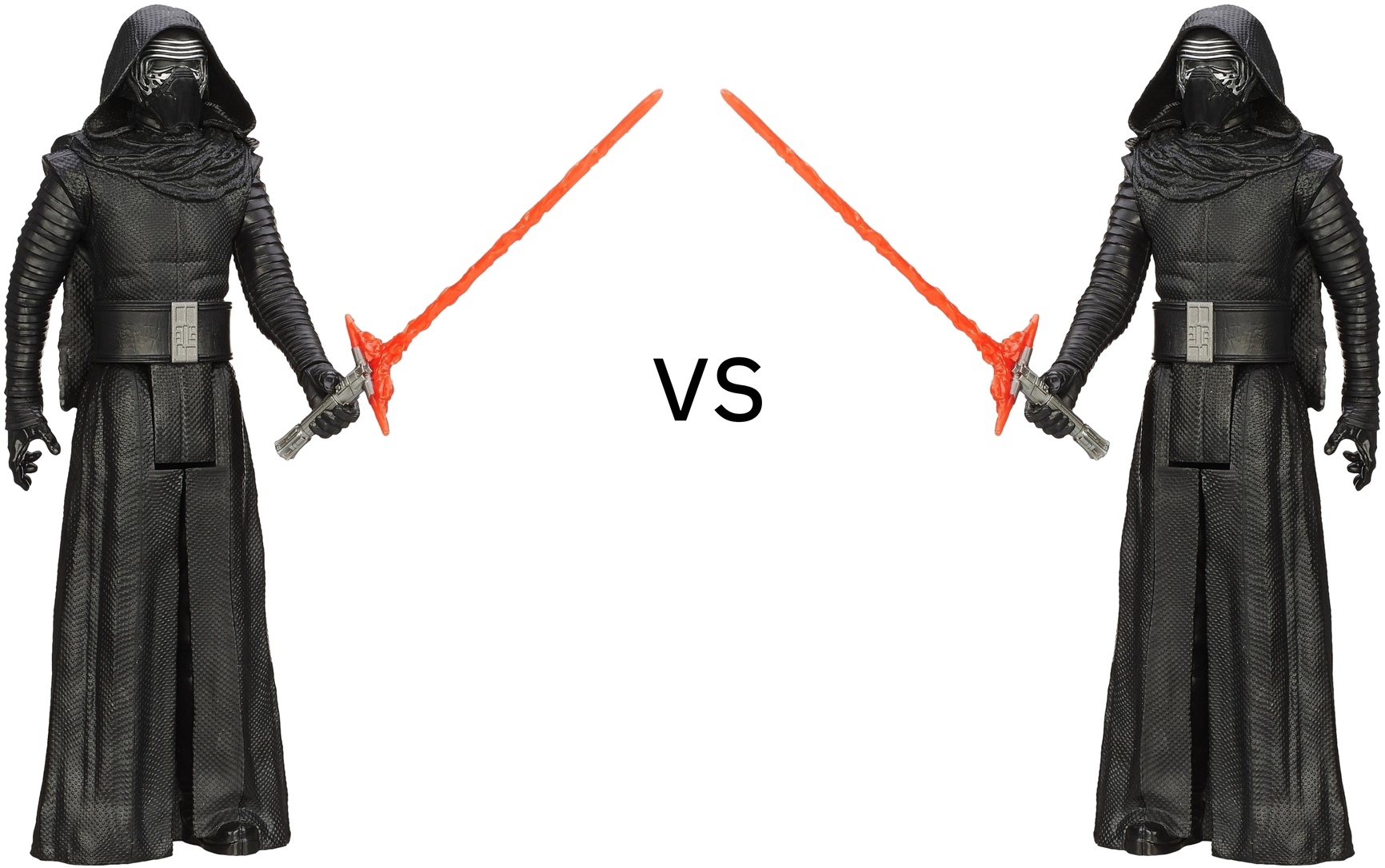
Nickolas Teixeira
@NTTL_LTTN

"The best technical tutorials are written by engineers who don't know the technology before writing them."

Thanks 🙏 @sehurlburt for the boost of confidence.

10:25 PM - Mar 20, 2018

424 83 people are talking about t


VS

— Generic!



— Flexible!



— Abstraction layer!



— UI widgets!

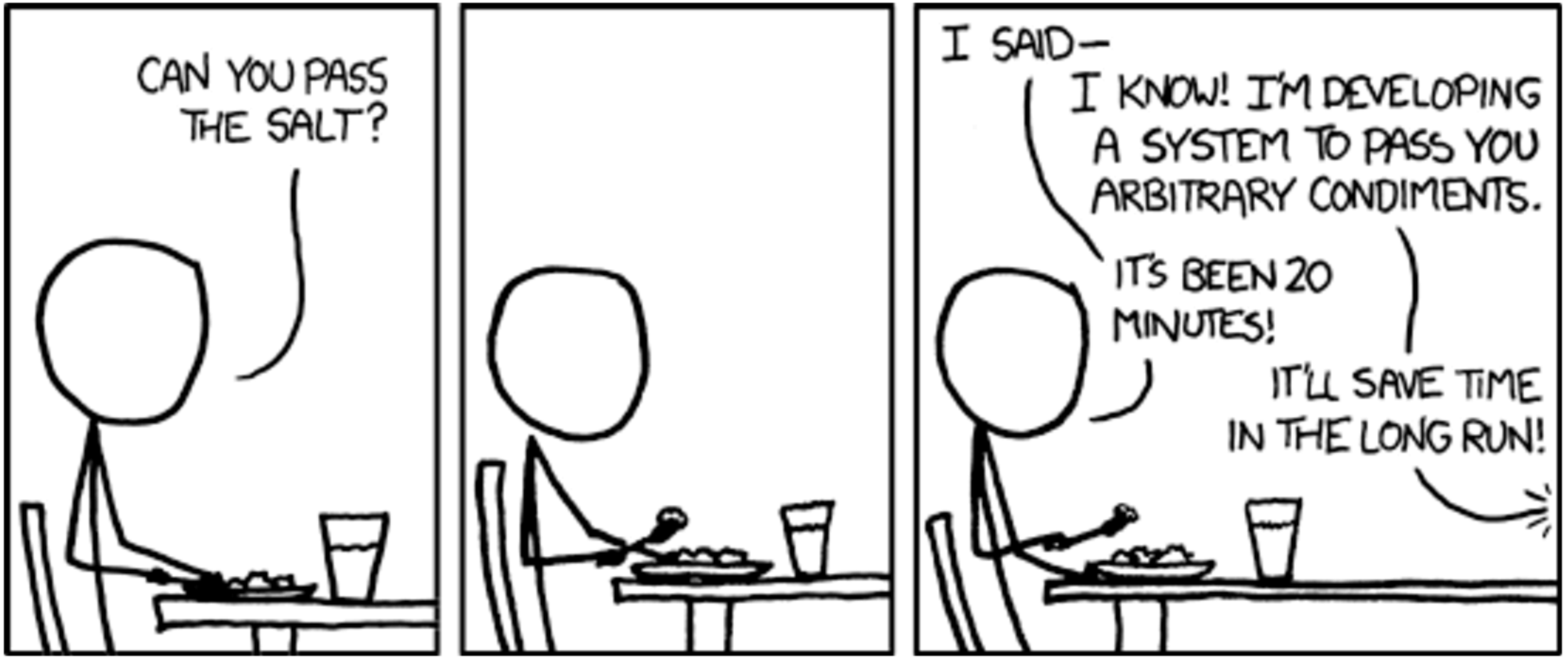


— Parser!



— ORM!




CAN YOU PASS THE SALT?
I SAID—
I KNOW! I'M DEVELOPING A SYSTEM TO PASS YOU ARBITRARY CONDIMENTS.
IT'S BEEN 20 MINUTES!
IT'LL SAVE TIME IN THE LONG RUN!



Eric Brandes

@BrandesEric

If a developer tells you they are building something "Generic" and "Flexible", rest assured that they're wasting your time and money.

7:21 PM - Jul 13, 2017

213 196 people are talking about t



VS





labs.ig.com/code-coverage-100-percent-tragedy

```
Scenario Outline: Get returns correct DtoToAdditionalDataModelMapper

  Given a DefaultDtoToAdditionalDataModelMapperRegistry configured with DtoToAdditionalDataModelMappers:
    | HiLoMarketDTO      | HiloBinaryMarketDtoToModelMapper      |
    | TunnelMarketDTO    | TunnelBinaryMarketDtoToModelMapper    |
    | OneTouchMarketDTO  | OneTouchBinaryMarketDtoToModelMapper  |
    | UpDownMarketDTO    | UpDownBinaryMarketDtoToModelMapper    |

  And a mock marketDto which returns <dtoName> from marketDto.getMarketDTOType()

  When DefaultDtoToAdditionalDataModelMapperRegistry getMapper(MarketDTO marketDto) is called with marketDto

  Then <dtoMapper> is the DefaultDtoToAdditionalDataModelMapperRegistry returned

Examples:
  | dtoName            | dtoMapper                             |
  | HiLoMarketDTO      | HiloBinaryMarketDtoToModelMapper      |
  | TunnelMarketDTO    | TunnelBinaryMarketDtoToModelMapper    |
  | OneTouchMarketDTO  | OneTouchBinaryMarketDtoToModelMapper  |
  | UpDownMarketDTO    | UpDownBinaryMarketDtoToModelMapper    |
```

```java
public class DtoToAdditionalDataModelMapperRegistryStepDefs {
    private DefaultDtoToAdditionalDataModelMapperRegistry testClass;
    private DtoToAdditionalDataModelMapper returned;
    private DtoToAdditionalDataModelMappersStepDefs additionalMapperTest = new DtoToAdditionalDataModelMappersStepDefs()
    private MarketDTO marketDTO;

    @Given("^a DefaultDtoToAdditionalDataModelMapperRegistry configured with DtoToAdditionalDataModelMappers:$")
    public void a_DefaultDtoToAdditionalDataModelMapperRegistry_configured_with_DtoToAdditionalDataModelMappers(Map<Stri
        Map<String, DtoToAdditionalDataModelMapper> registry = new HashMap<>();

        for (Map.Entry<String, String> entry : mapOfMappers.entrySet()) {
            DtoToAdditionalDataModelMapper mapper = createInstanceOf(PACKAGE, entry.getValue());
            registry.put(entry.getKey(), mapper);
        }

        testClass = new DefaultDtoToAdditionalDataModelMapperRegistry(registry);
    }

    @Given("^a mock marketDto which returns ([^\\s]+) from marketDto.getMarketDTOType\\(\\)$")
    public void a_mock_marketDto_which_returns_from_marketDto_getMarketDTOType_(String type) throws Throwable {
        marketDTO = Mockito.mock(MarketDTO.class);
        when(marketDTO.getMarketDTOType()).thenReturn(type);
    }


    @When("^DefaultDtoToAdditionalDataModelMapperRegistry getMapper\\(MarketDTO marketDto\\) is called with marketDto$")
    public void DefaultDtoToAdditionalDataModelMapperRegistry_getMapper_MarketDTO_marketDto_is_called_with_marketDto() t
        returned = testClass.getMapper(marketDTO);
    }

    @Then("^([^\\s]+) is the DefaultDtoToAdditionalDataModelMapperRegistry returned$")
    public void _dtoMapper_is_the_DefaultDtoToAdditionalDataModelMapperRegistry_returned(String expectedMapper) throws T
        assertEquals(expectedMapper, returned.getClass().getSimpleName());
    }

}
```

```java
public class TestClassLoader {

    public static <T> T createInstanceOf(String className, Object... initArgs) {
        try {
            Class clazz = Class.forName(className);
            Constructor constructor = clazz.getDeclaredConstructors()[0];
            return (T)  constructor.newInstance(initArgs);
        } catch (ClassNotFoundException | InstantiationException | IllegalAccessException |
            throw new RuntimeException("Could not create and instance of class: " + classNam
                    "does it have a default constructor? is it public?", e);
        }

    }

    public static <T> T createInstanceOf(String packageName, String className, Object... i
        return createInstanceOf(packageName + className, initArgs);
    }
}
```

```java
public class DefaultDtoToAdditionalDataModelMapperRegistry implements DtoToAddi

    private final Map<String, DtoToAdditionalDataModelMapper> registry;

    public DefaultDtoToAdditionalDataModelMapperRegistry(Map<String, DtoToAdditi
        registry = new HashMap<>(pRegistry);
    }

    @Override
    public DtoToAdditionalDataModelMapper getMapper(MarketDTO marketDto) {
        return registry.get(marketDto.getMarketDTOType());
    }

}
```

“That is a big waste of time.”

“But my boss expects me to write tests for all classes.”

“At the expense of?”

“Expense?”

“Anyway, those tests have nothing to do with BDD.”

“I know, but we decided to use Cucumber for all tests.”

“😕”

“😕”

Методика: тесты помогают повысить стабильность системы к изменениям



Методика: тесты помогают повысить стабильность системы к изменениям



Методика: тесты помогают повысить стабильность системы к изменениям





Методика: тесты ~~помогают повысить стабильность системы к изменениям~~





Методика: тесты



Правило: Тесты — хорошо, больше тестов — еще лучше.
Пишем 100% тестов



VS



- Нужна разница в уровнях
- Лучший специалист не всегда лучший учитель
- Удачные и неудачные комбинации

Книги







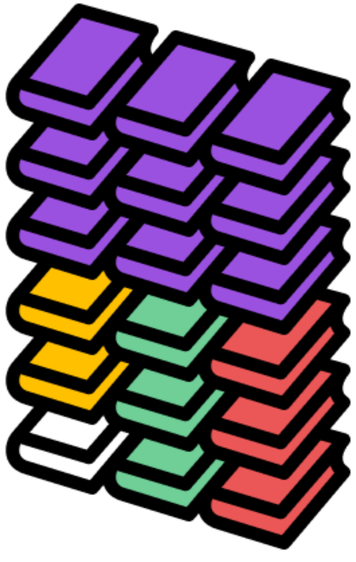




























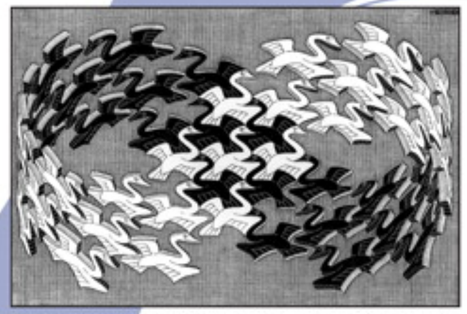


1. Правила
2. Рекомендации
3. Методики
4. Максимы
5. ???

Design Patterns

Elements of Reusable Object-Oriented Software

Erich Gamma
Richard Helm
Ralph Johnson
John Vlissides



Cover art © 1994 M.C. Escher / Cordon Art - Baarn - Holland. All rights reserved.

Foreword by Grady Booch

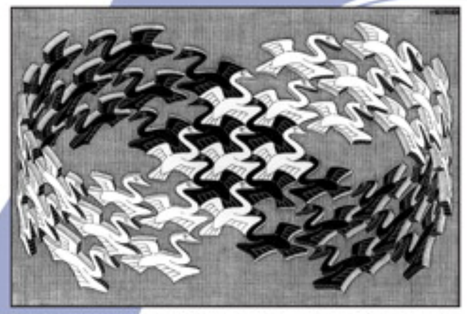
ADDISON-WESLEY PROFESSIONAL



Design Patterns

Elements of Reusable Object-Oriented Software

Erich Gamma
Richard Helm
Ralph Johnson
John Vlissides



Cover art © 1994 M.C. Escher / Cordon Art - Baarn - Holland. All rights reserved.

Foreword by Grady Booch

ADDISON-WESLEY PROFESSIONAL COMPUTING SERIES





Zach Holman
@holman

Follow

1. Try something new
2. Read all the advice from experts
3. Nod and say “sure, no problem”
4. Make all the mistakes anyway
5. Finally understand for yourself what that advice really meant
6. Advise beginners not to do that thing
7. Repeat

2:27 AM - 29 Apr 2018

326 Retweets **1,090** Likes

14 326 1.1K

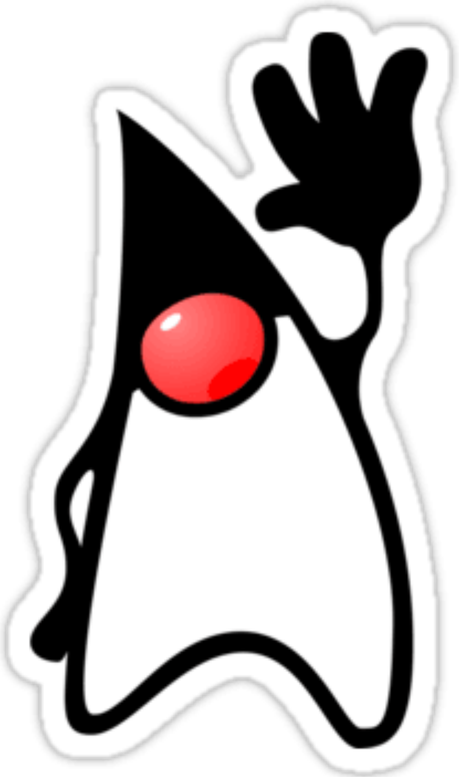


tema (tema) wrote,
2009-05-05 17:31:00

> Обнаружил эффект: когда в какой-либо области начинаешь хорошо разбираться (насколько это вообще возможно в моем случае), в какой-то момент начинает от книг по теме тошнить. Скажем, я вообще не могу читать о логотипах. О шрифтах скоро перестану мочь.

https://tema.livejournal.com/341629.html

Языки



Классы, интерфейсы, иерархии, правила наследования, правила видимости, вложенные классы, исключения, статические методы



- Тоже ООП
- Динамический
- Соглашения вместо ограничений
- Monkey patching
- Максимы!



PEP-20 Zen of Python

— Beautiful is better than ugly
— Explicit is better than implicit
— Simple is better than complex
— Complex is better than complicated
— Flat is better than dense
— Readability counts
— Special cases aren’t special enough to break the rules
— Although practicality beats purity
— Errors should never pass silently
— Unless explicitly silenced

— In the face of ambiguity, refuse the temptation to guess
— There should be one—and preferably only one—obvious way to do it
— Although that way may not be obvious at first unless you’re Dutch
— Now is better than never
— Although never is often better than *right* now
— If the implementation is hard to explain, it’s a bad idea
— If the implementation is easy to explain, it may be a good idea



Do it yourself

Классов нет. Типов нет. ООП нет. Ничего нет.

Все доступно и открыто всем.

Все можно переопределить

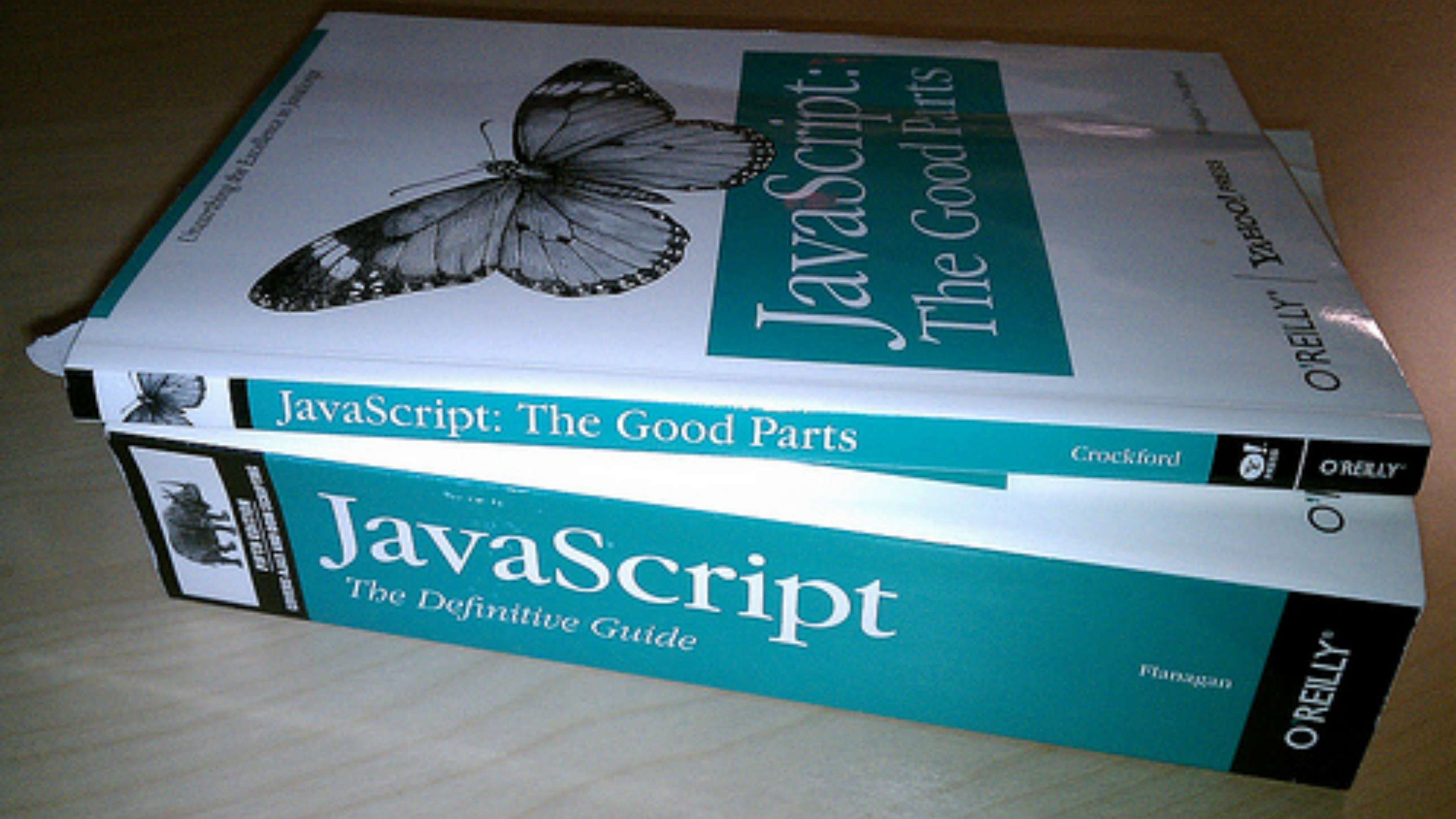
Даже синтаксис

— **Can we make programs out of simpler stuff**? That's the problem after 18 years of using, like, C++ and Java, **you're exhausted**.

How many people have been programming for 18 years? How many for more than 20? More than 25? Less than 5? [...]

It may be an indictment of Clojure as a beginner's language or may be that **Clojure is the language for cranky, tired, old programmers**


Пишите как хотите —


JavaScript: The Good Parts
O'REILLY
JavaScript: The Good Parts
Crockford
O'REILLY
JavaScript
The Definitive Guide
Flanagan
O'REILLY

- Ужесточение конструкций (prototypes → classes)
- Огромная библиотека тривиальных функций (npm)

Заключение

- Более глубокое понимание причин
- Обучение, рост, общение, споры
- Книги, лекции, проекты, языки

Дискасс!

@nikitonsky