Apache ECharts (incubating)

Web-based Interactive Big Data Visualization

Wenli Zhang

ApacheCon North America 2019

How Big?

10,000,000

data pieces

1920x1080

= 2,073,600 pixels

So...

10,000,000

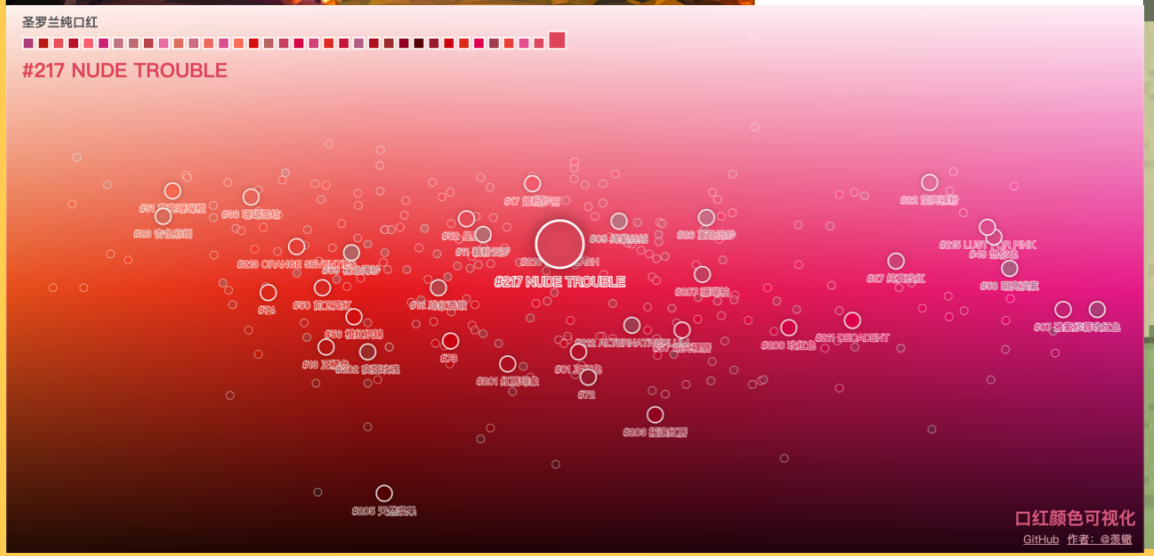
data pieces?

Full HD

About Me

- Wenli Zhang
- Shanghai, China
- PPMC of Apache ECharts (incubating)
- Joined Baidu (China) Co., Ltd. in April 2016
  - Senior R&D Engineer

github.com/Ovilia



Apache ECharts (incubating)

A JavaScript Visualization Tool And More...

Milestones

Oct. 2012

Project Started by @kener inside Baidu (China) Co., Ltd.

Jun. 2013

Open Sourced; Released **v1.0**

Jun. 2014

Released **v2.0**

Jan. 2016

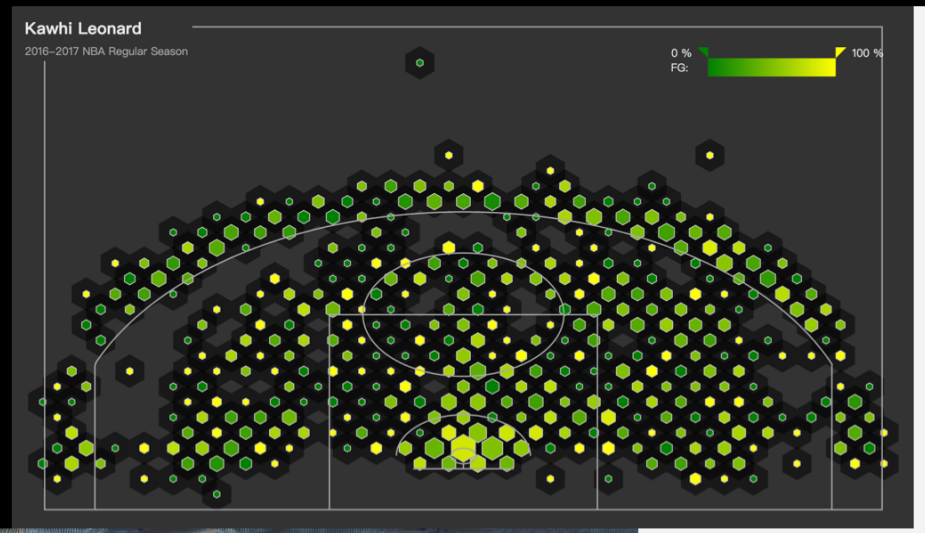
Released **v3.0**

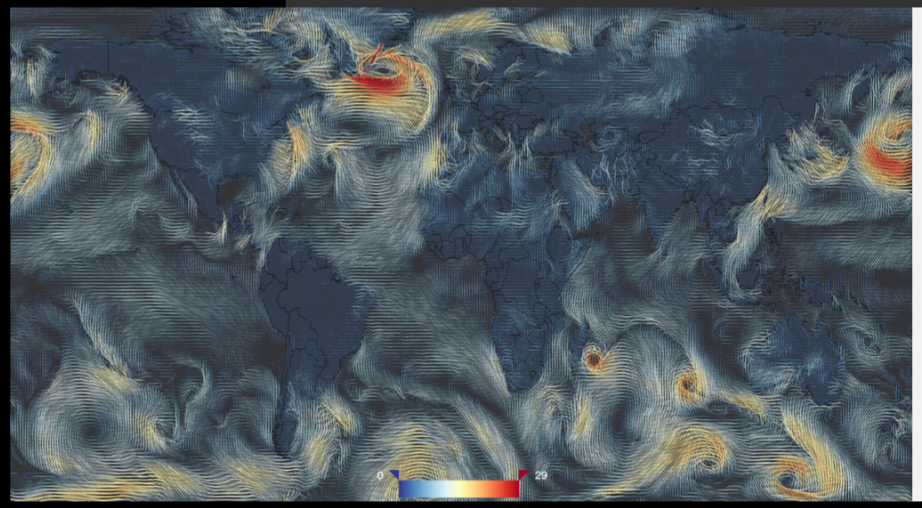
Jan. 2018

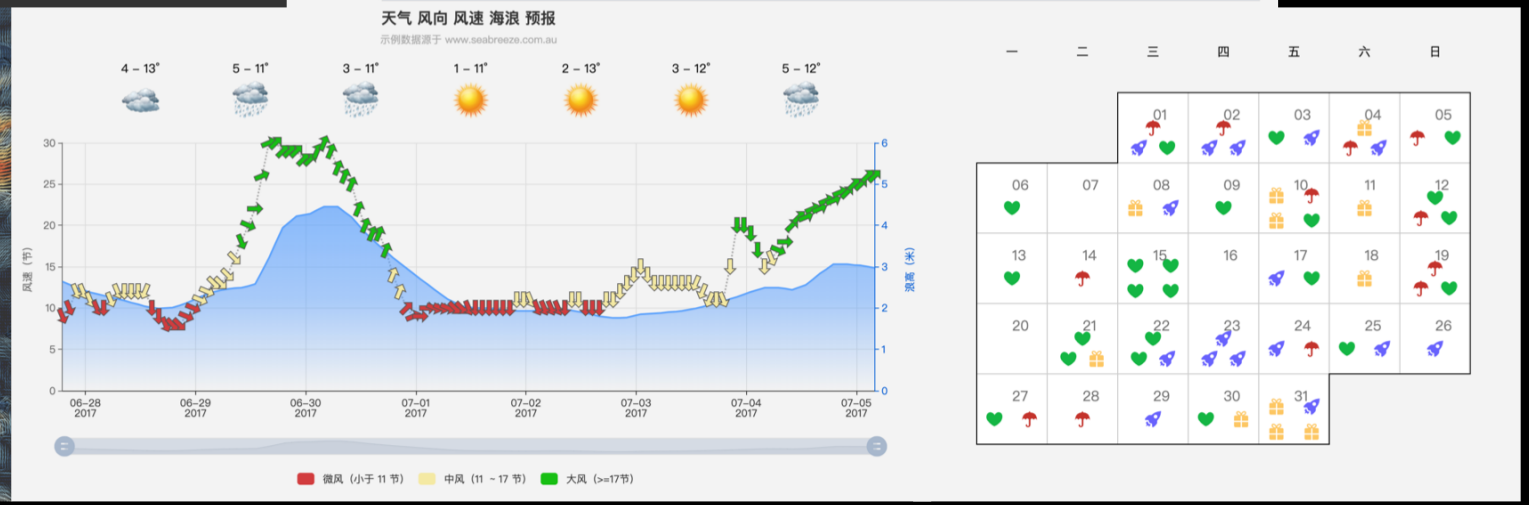
Start incubating at the ASF; Released **v4.0**

Feature 1: Extensive Chart Types

Examples of Custom Charts







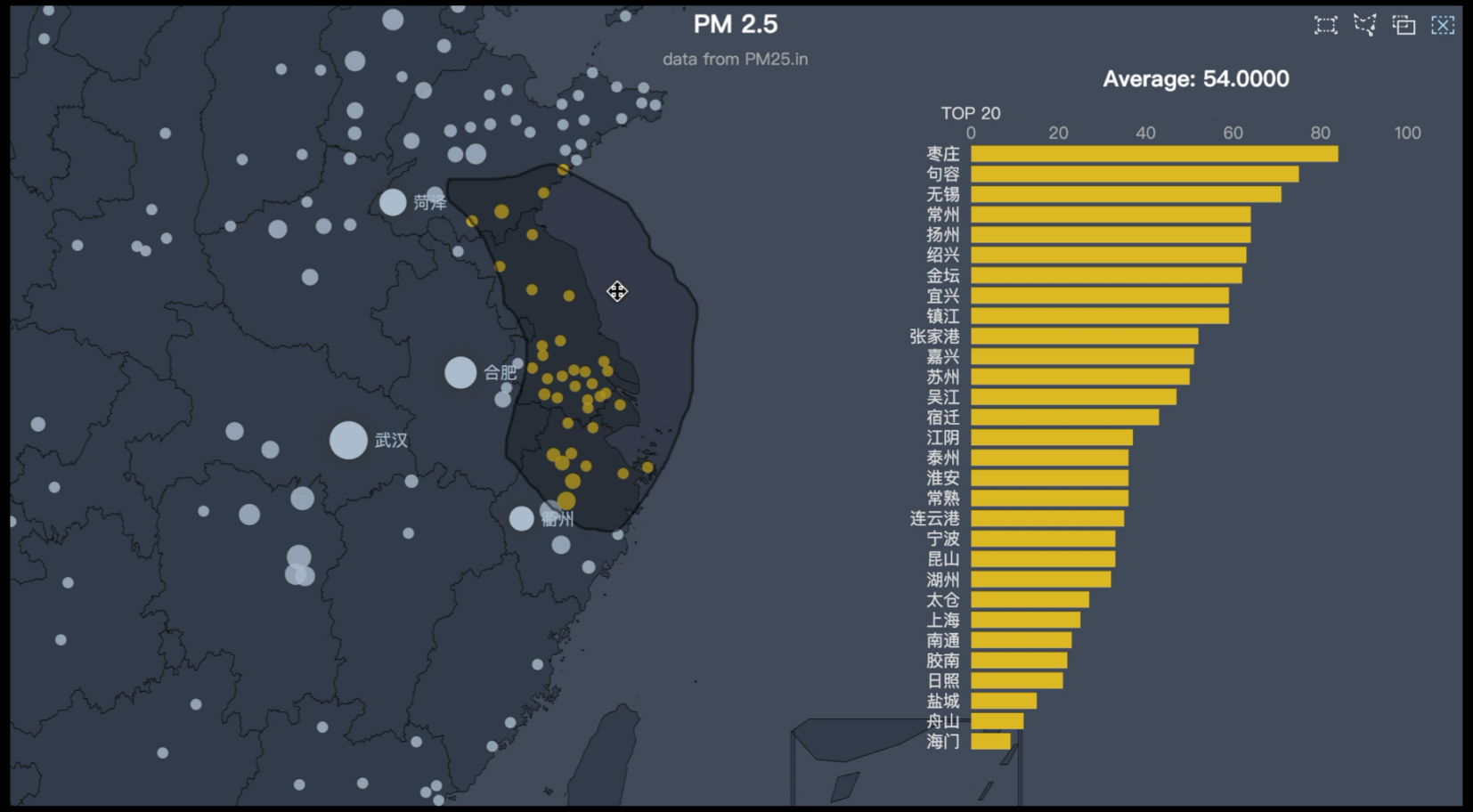
Feature 2: Cross-platform Solution

- Rendering Mechanism
  - Canvas
  - SVG
  - VML
- Responsive Design
  - PC
  - Mobile
- Other Languages (from community)
  - Node.js
  - Python
  - R
  - ……

Responsive Design

Feature 3: Powerful Interactions

- Legend
- Visual Mapping
- Data Zooming
- Tooltip
- Brushing
- ...



Example of Brushing

Feature 4: Visualizing 10,000,000s Data

Rendering Big Data

Why front-end?

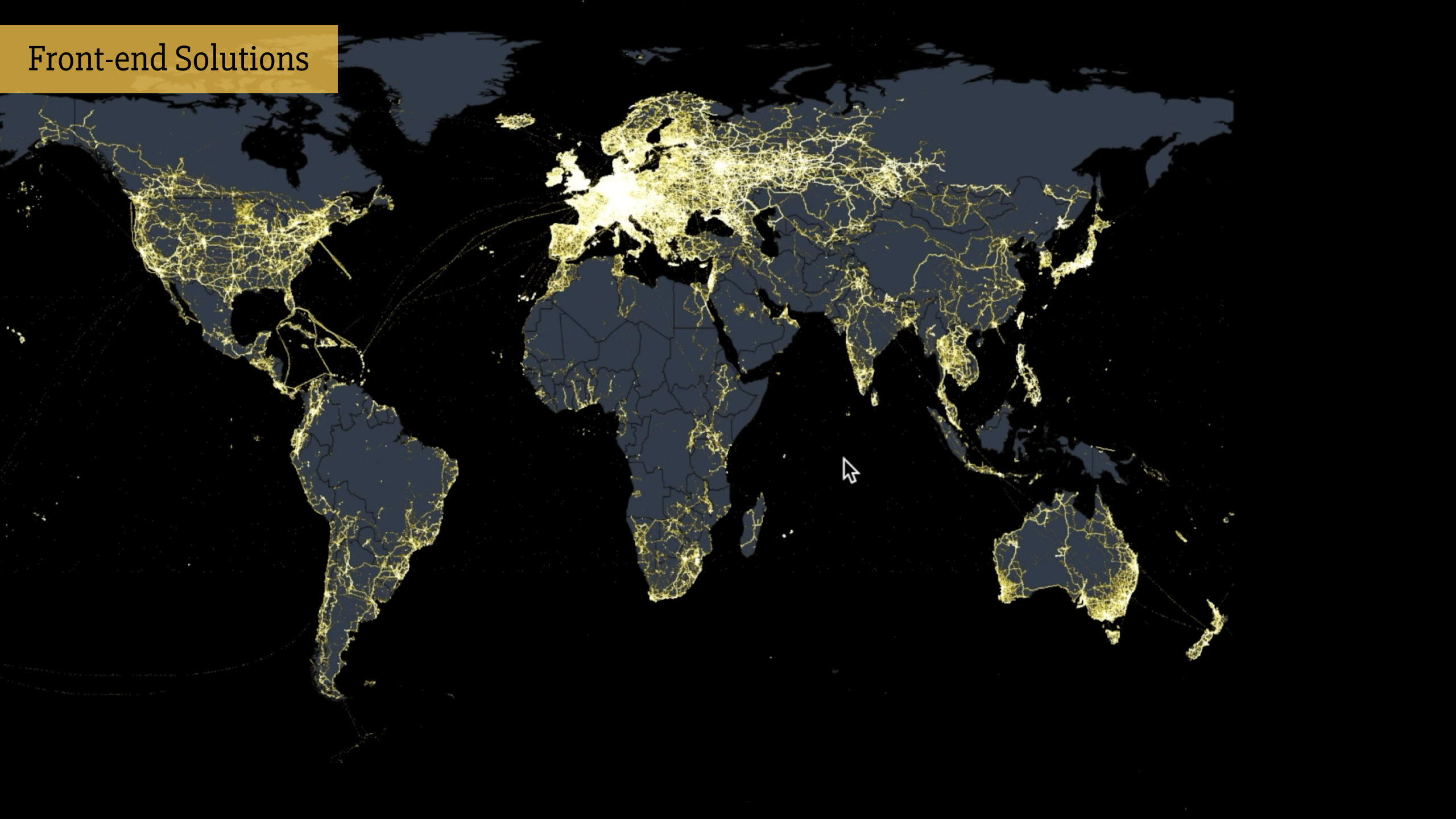
Back-end Solutions

Plan A: Generate images

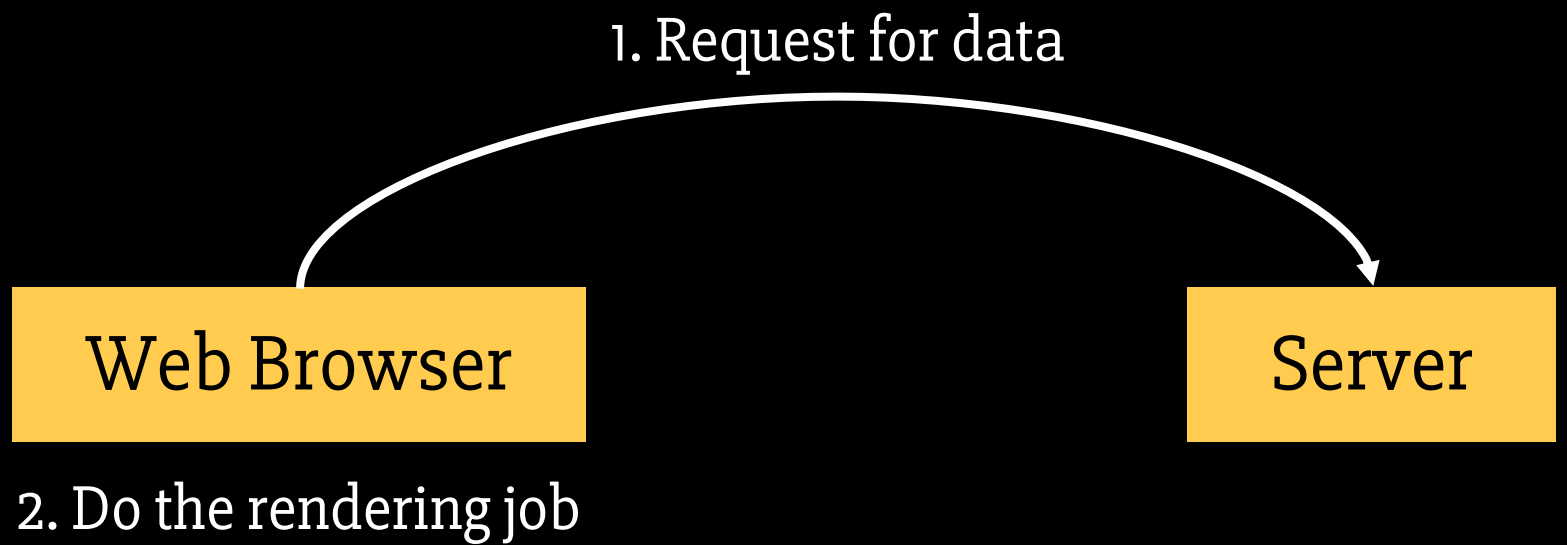
Plan B: Pre-calculate data

- Bad Interaction
- Bad Visual Result
- Extra Coding
- Extra Computing Resources

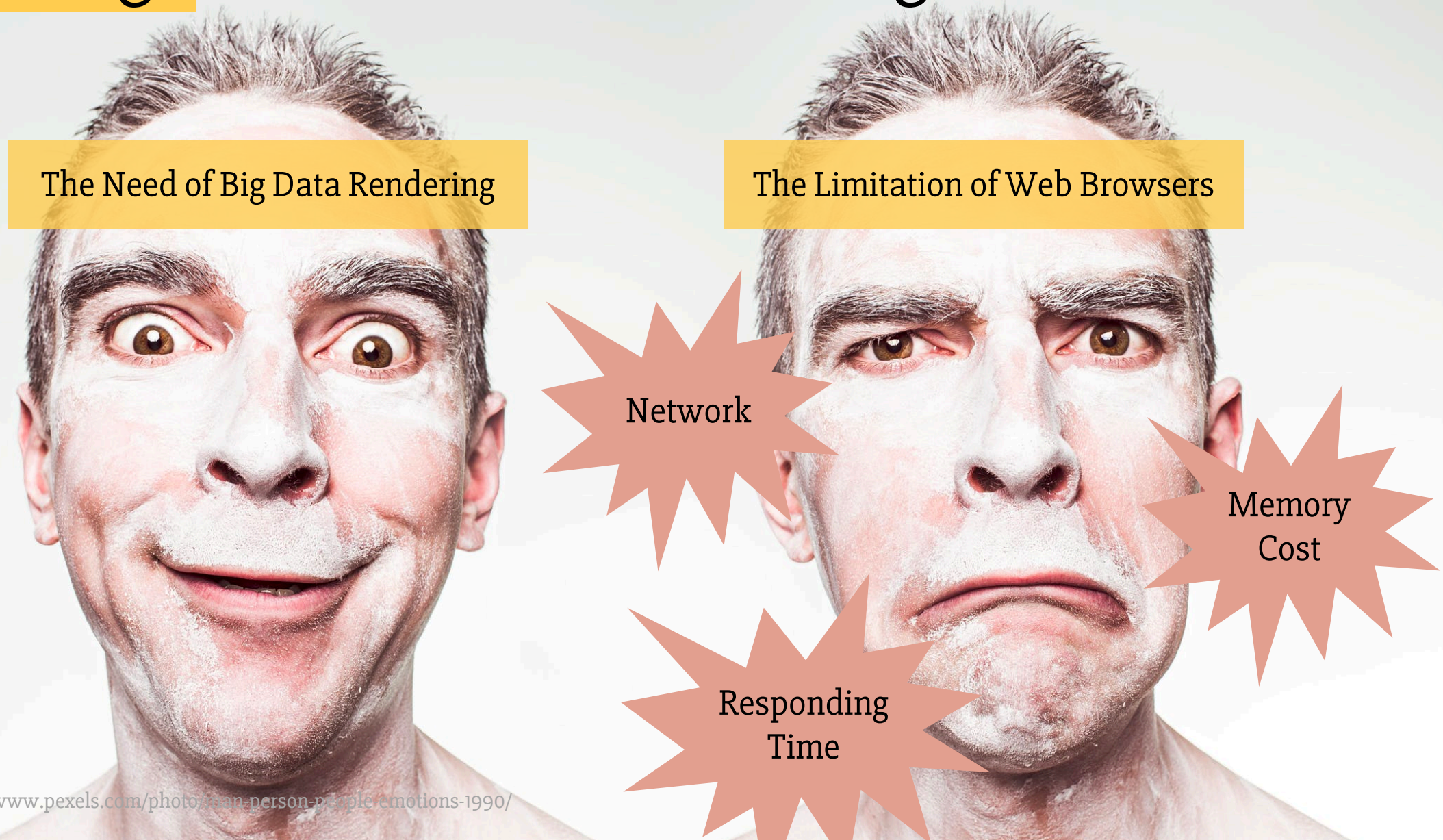
Front-end Solutions



Basic Workflow of Web-based Visualization



Challenge of Front-end Rendering



The Need of Big Data Rendering

The Limitation of Web Browsers

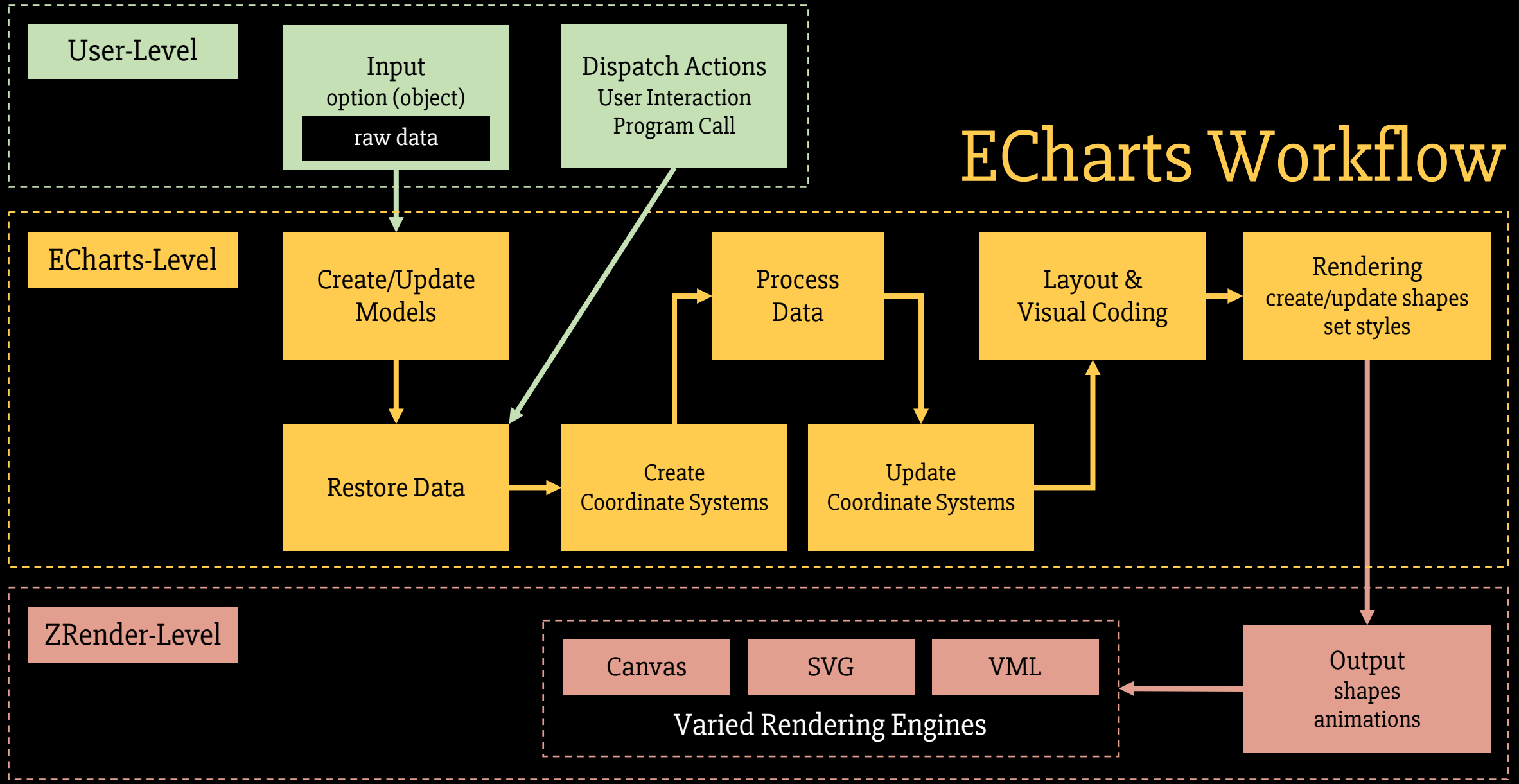
Network

Memory Cost

Responding Time

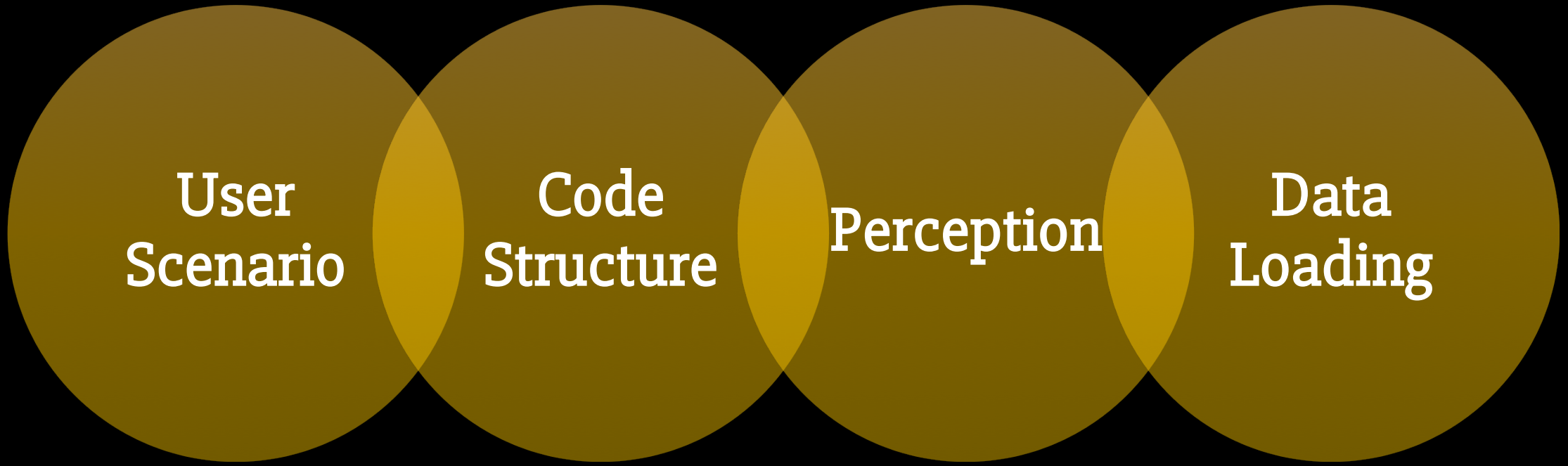
Image Source: https://www.pexels.com/photo/man-person-people-emotions-1990/

ECharts Workflow

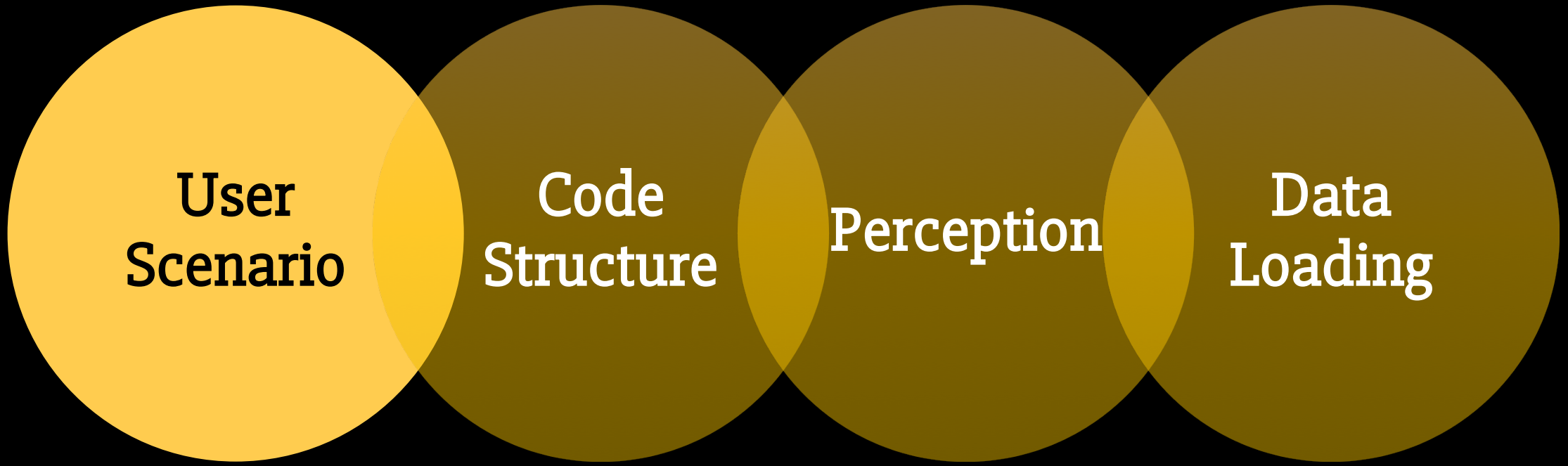


Li, Deqing & Mei, Honghui & Shen, Yi & Su, Shuang & Zhang, Wenli & Wang, Junting & Zu, Ming & Chen, Wei. (2018). ECharts: A declarative framework for rapid construction of web-based visualization. Visual Informatics. 2. 10.1016/j.visinf.2018.04.011.

ECharts' Solution



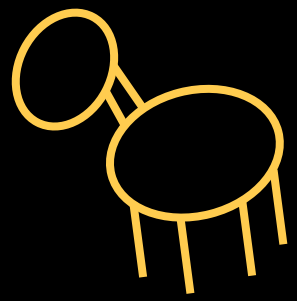
ECharts' Solution



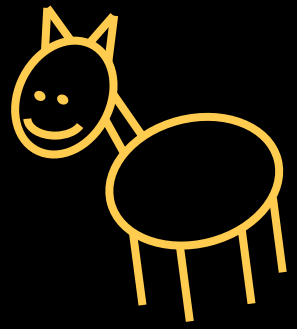
Solution 1: User Scenario Based Approach

1. Find the bottleneck of a given scenario
2. Enhance it!

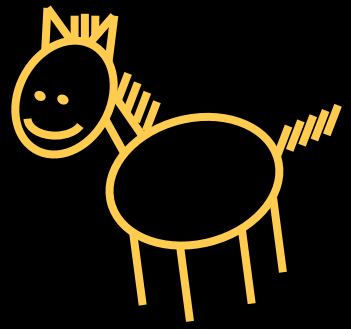
Step 1.
Draw 2 circles



Step 2.
Draw 6 lines



Step 3.
Draw the face

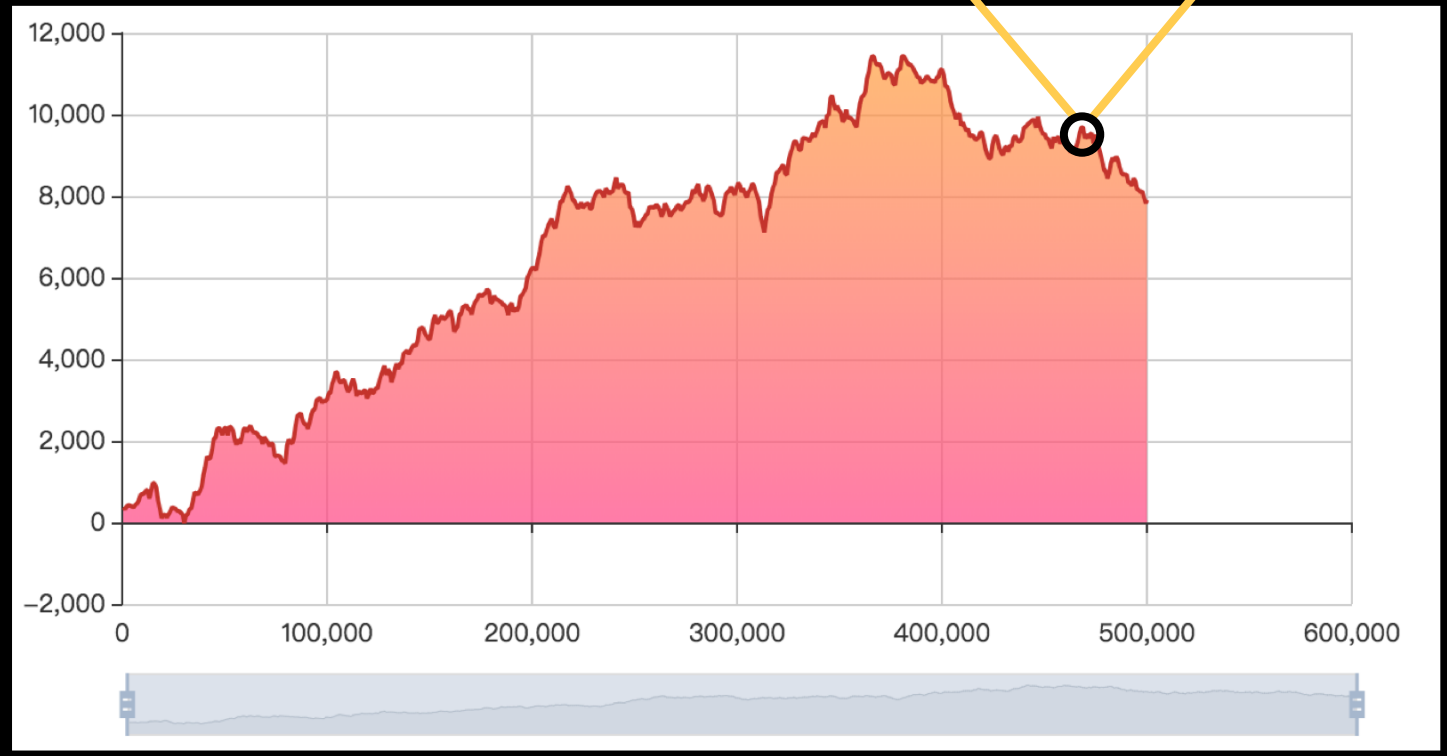


Step 4.
Add hair

Step 5.
Add more details

Case 1: Line chart with 500,000 data

- Downsampling

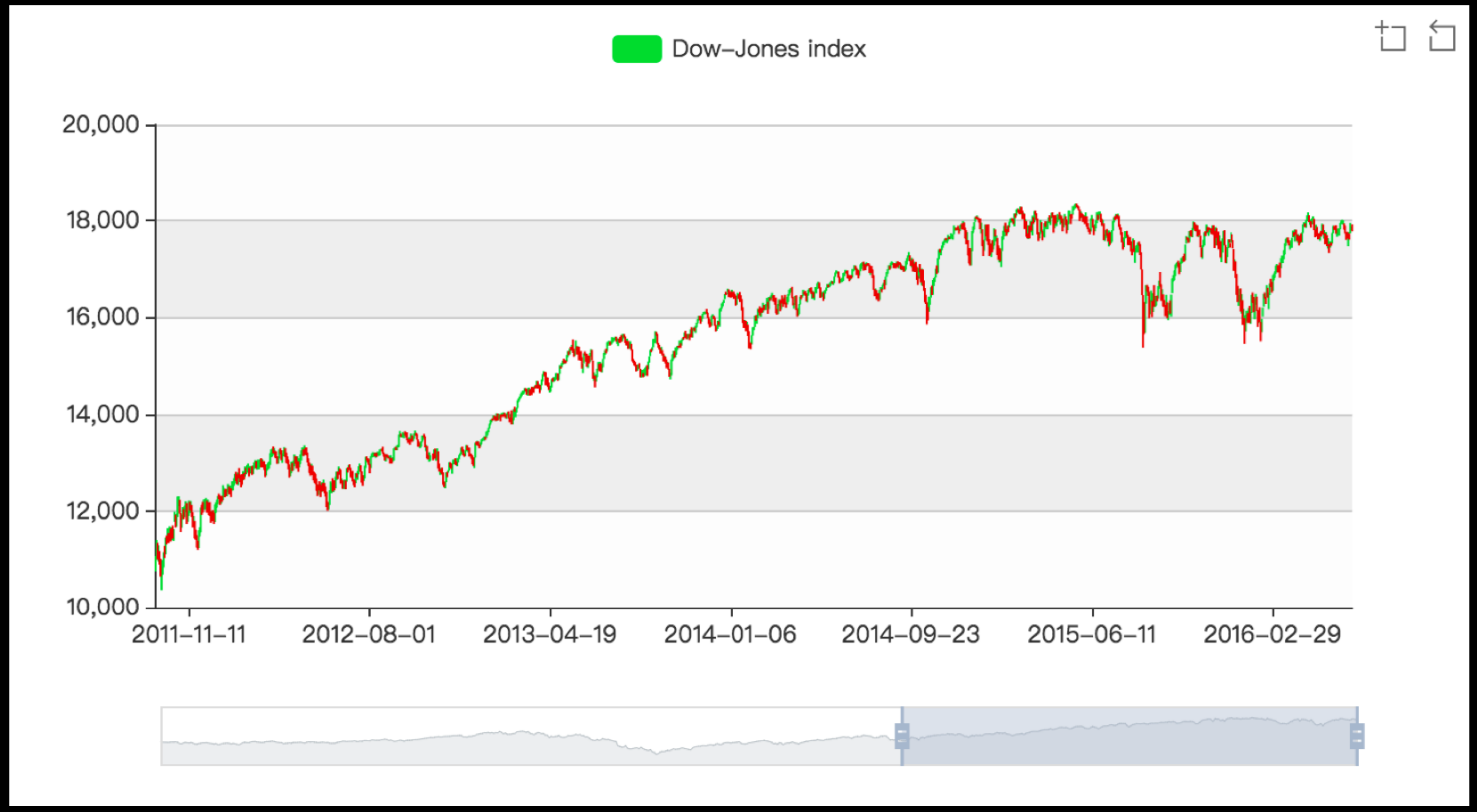


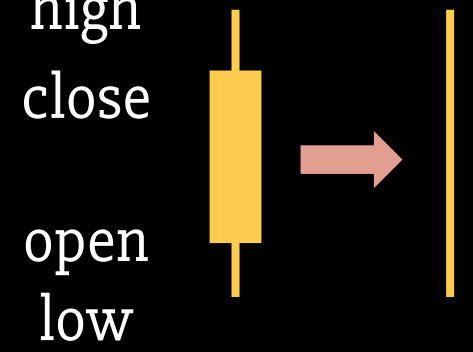
500 data
in 1 pixel!

grid width: 1000px

Case 2: Candlestick chart with big data

- Simplify the shape





Pros & Cons
of User Scenario Based Approach

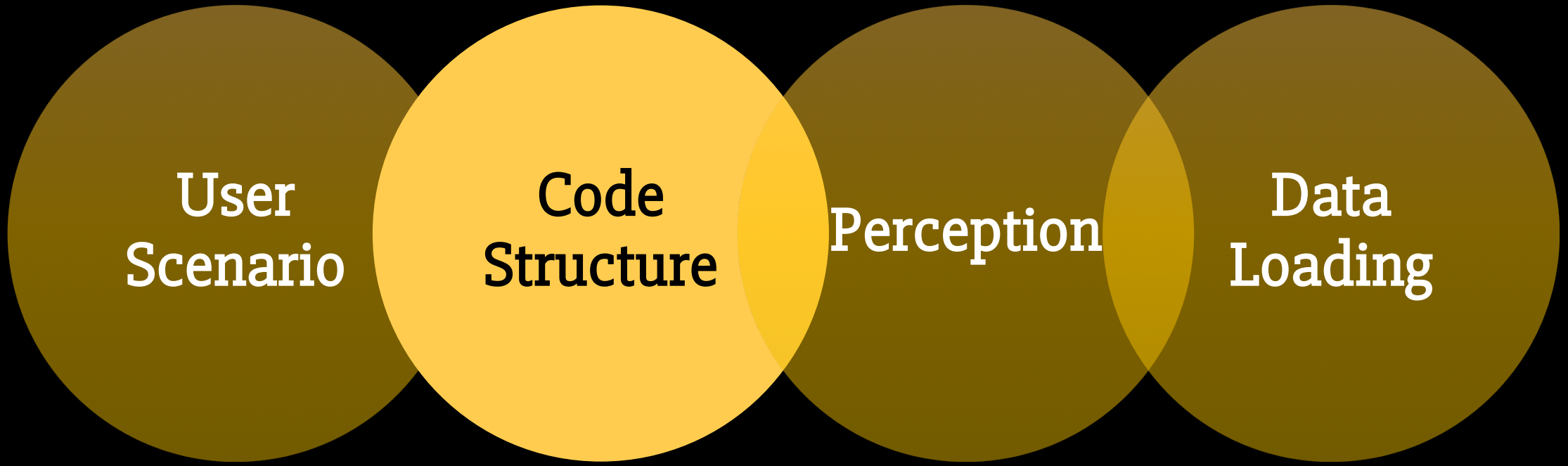
Pros

- Magnificent improvement
- Low cost of development
- Low impact on the visual result

Cons

- Limited to certain scenarios

ECharts' Solution



Solution 2: Code Structure Based Approach

- Change the code structure or code expressions
- To enhance the efficiency

Shapes with the same style

```
ctx.fillStyle = ...
ctx.rect(...);
ctx.fillStyle = ...
ctx.rect(...);
ctx.fillStyle = ...
ctx.rect(...);
ctx.fillStyle = ...
ctx.rect(...);
```

Before

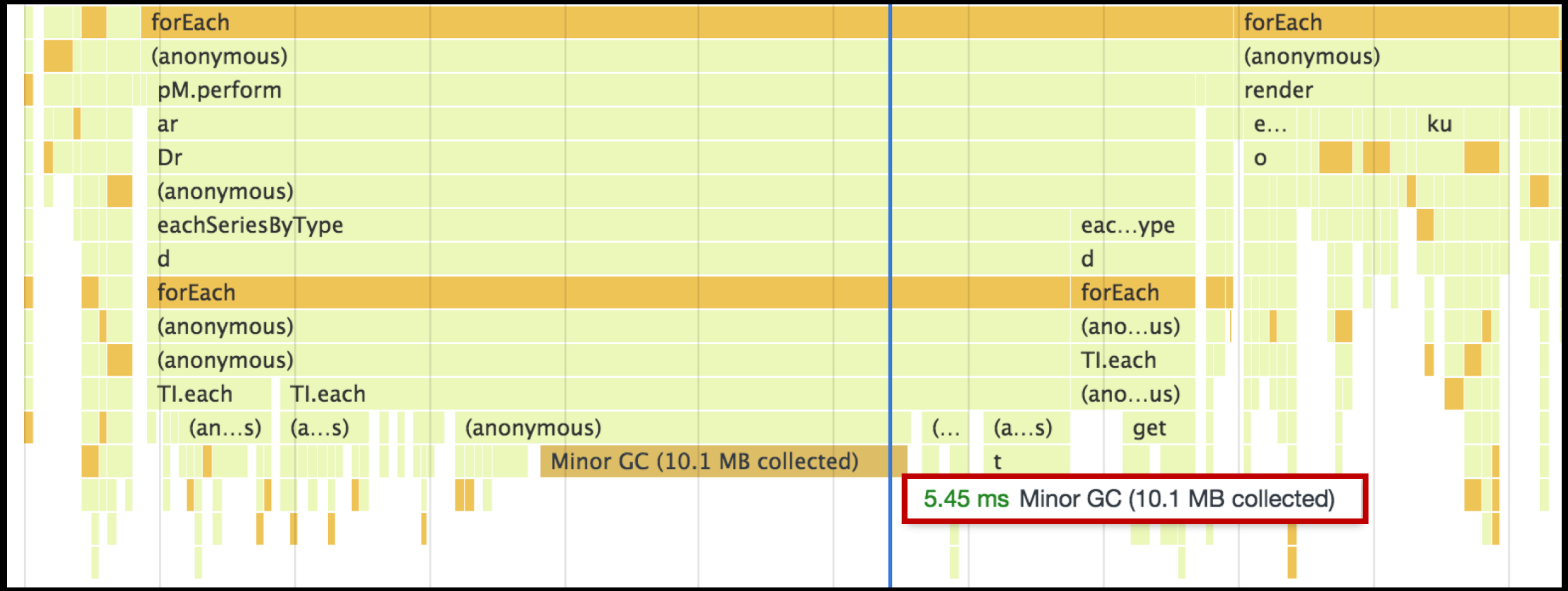
Render a rectangle each

```
ctx.fillStyle = ...
ctx.rect(...);
ctx.rect(...);
ctx.rect(...);
ctx.rect(...);
```

After

Render compound graphic element

Garbage Collection



The Cost of 5 milliseconds

- 60FPS: 1000ms / 60 = 16ms
- 20FPS: 1000ms / 20 = 50ms

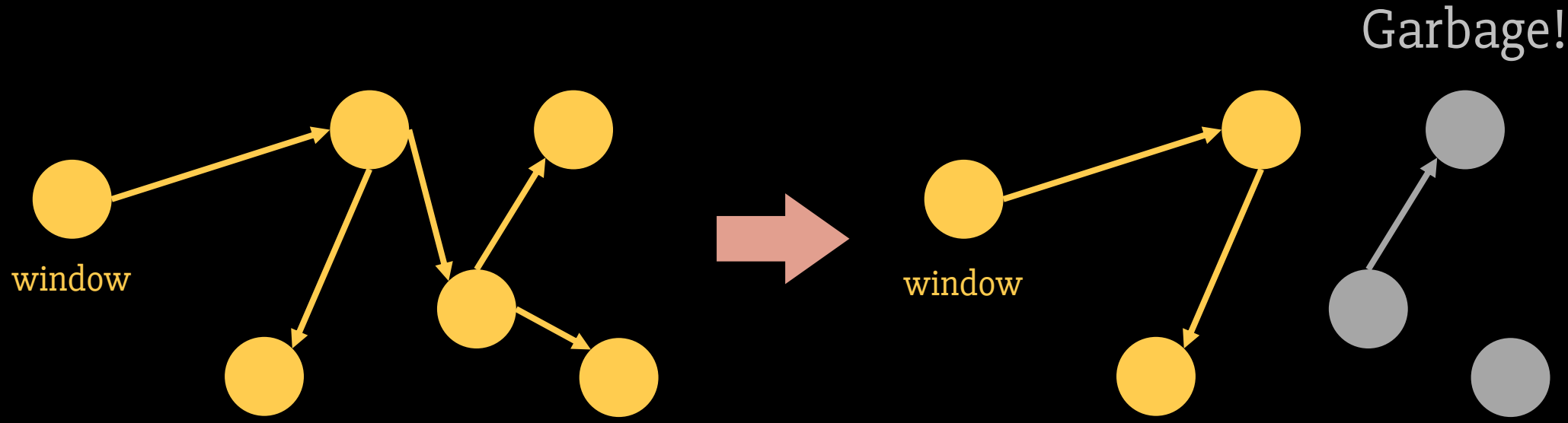
We don’t have time for

GARBAGE COLLECTION!

What is **Garbage**?

What is **Garbage**?

- Objects that cannot be reached any more



At first...

After some while...

When to do Garbage Collection?

- No space for new allocation

How to do Garbage Collection?

Object Lifetime

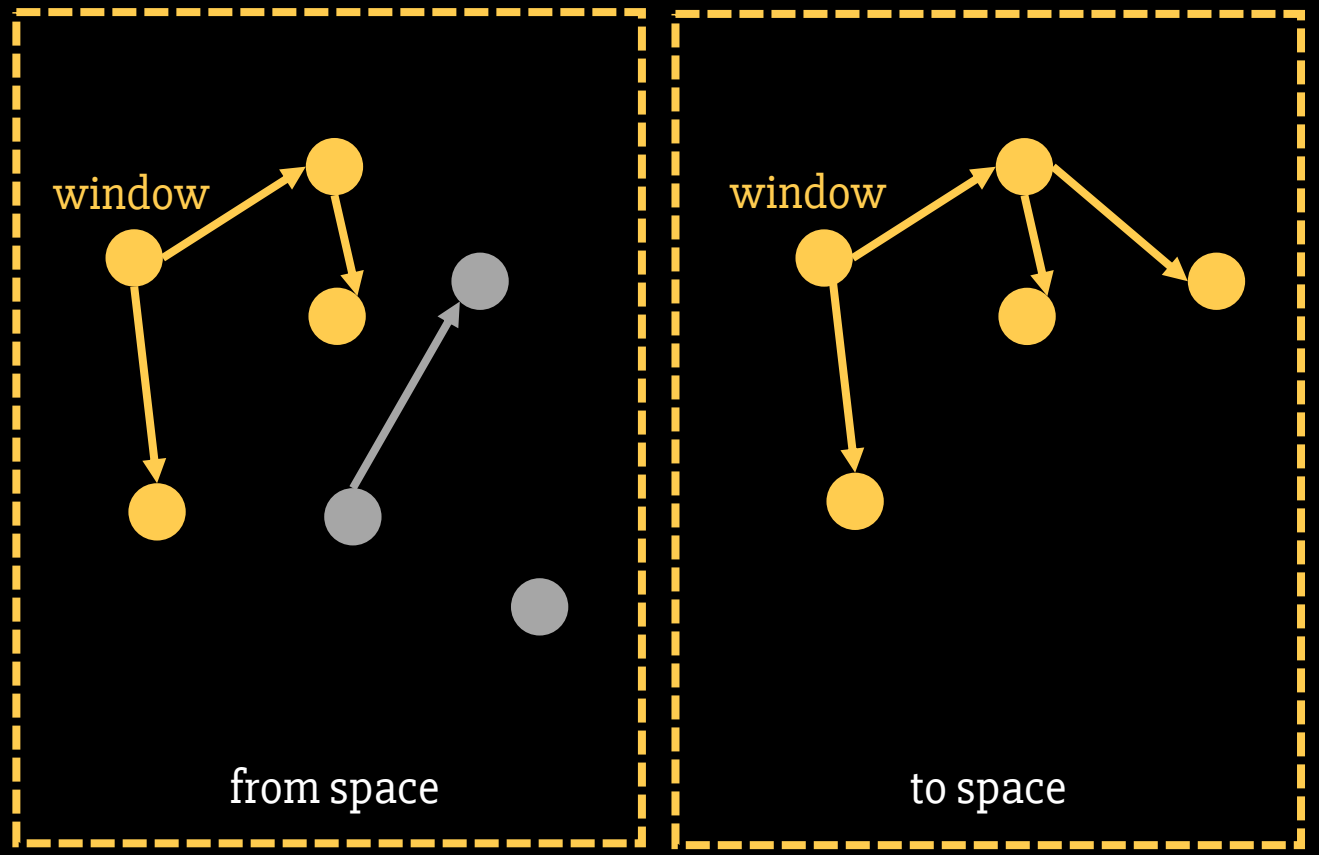
- Most of them
  - Die young
  - Occupy limited space
- The rest
  - Live a long life
  - Occupy more space

The Young Generation

The Old Generation

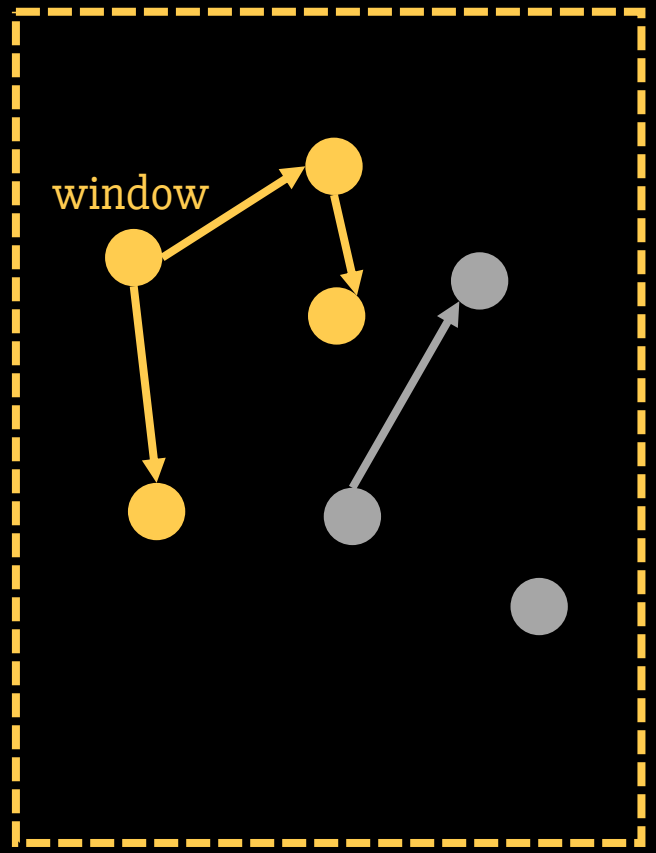
The Young Generation

- Feature
  - Die young
  - Occupy limited space
- Policy: Scavenge
  - Space-for-Speed
  - 2x space



The Old Generation

- Feature
  - Live a long life
  - Occupy more space
- Policy: Mark-Sweep/Compact GC
  - which takes a longer time



Further reading: https://github.com/joyeecheung/talks/blob/master/node_interactive_2016/are_your_v8_garbage_collection_logs_speaking_to_you_updated.pdf

How to save GC time?

What's the most time-consuming part in GC?

Large Objects of Old Generation

Avoid getting old

- Release early
- Re-use objects

Avoid being large

- Using typed arrays
  - Reduce memory cost
- Avoid re-allocation
- Avoid sparse array

Pros & Cons
of Code Structure Based Approach

Pros

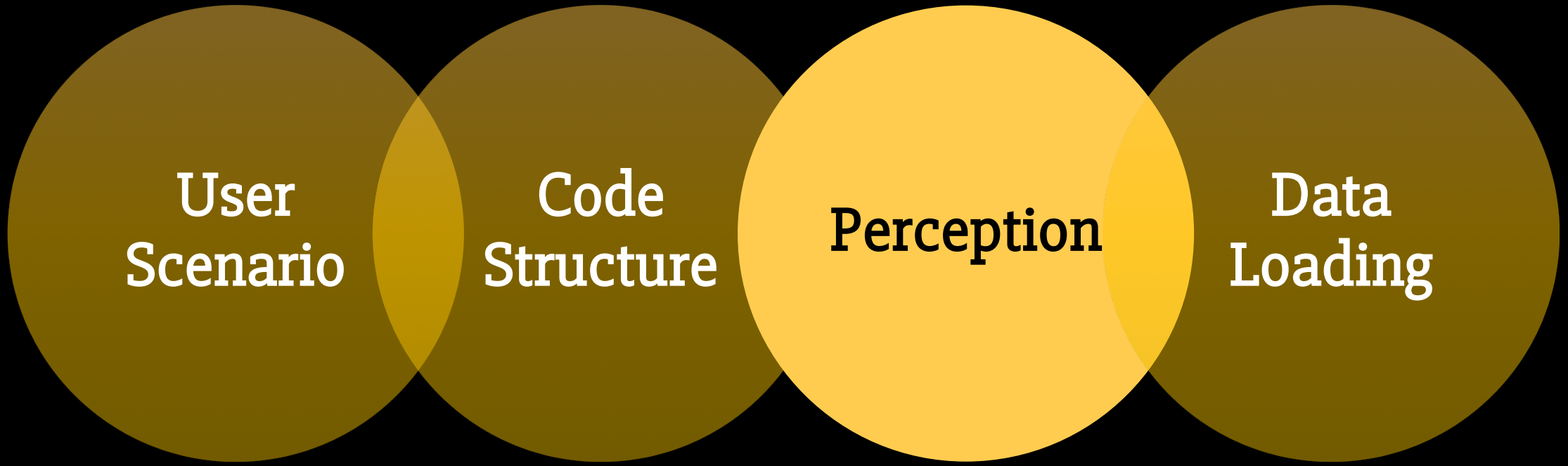
- Improves to some extent

Cons

- Damage
  - code structure
  - readability
  - robustness

Use only when necessary!

ECharts' Solution



If we can’t speed up any more,
at least we could...

LET THE VISITORS

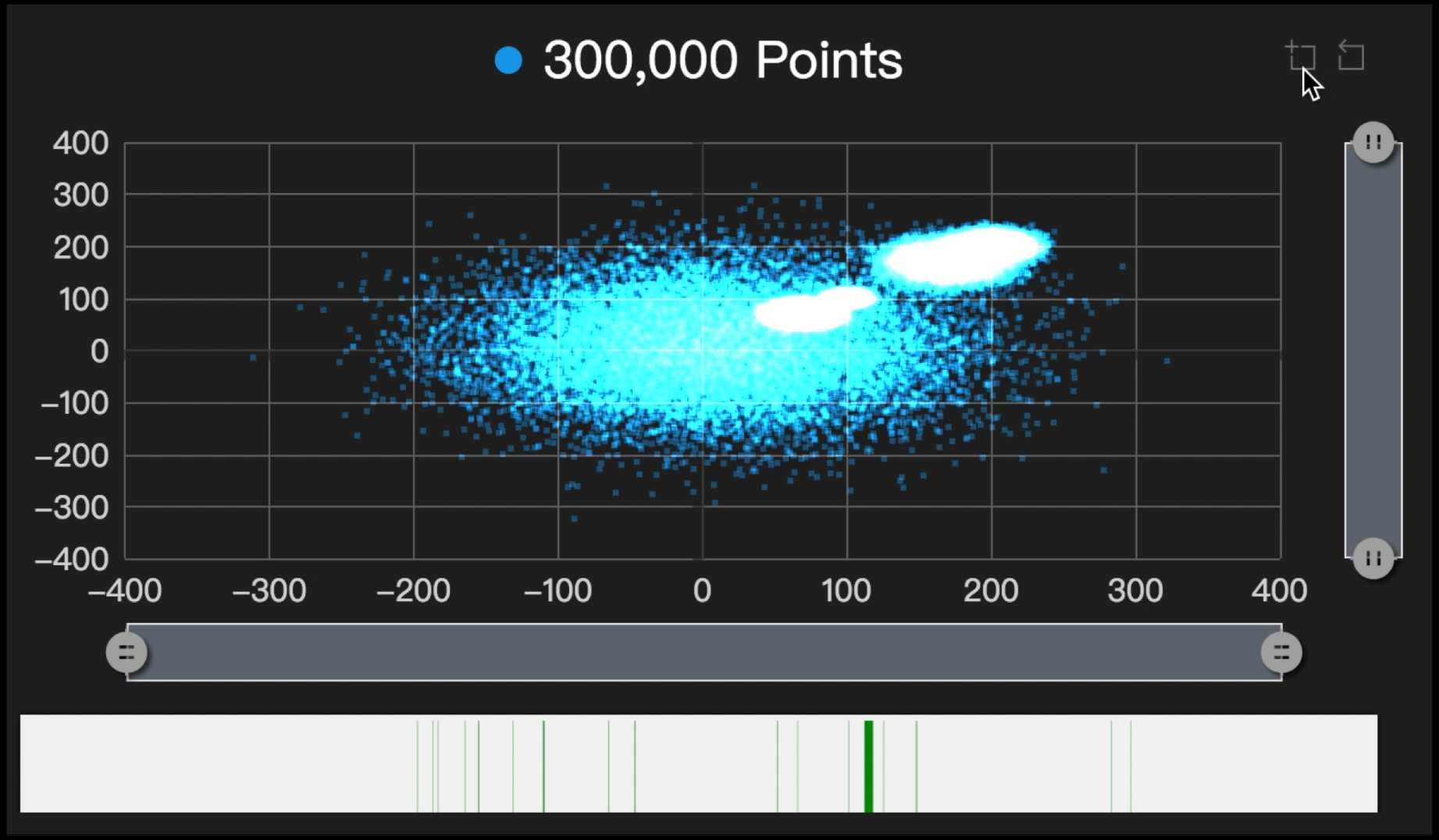
BELIEVE

IT’S FAST!

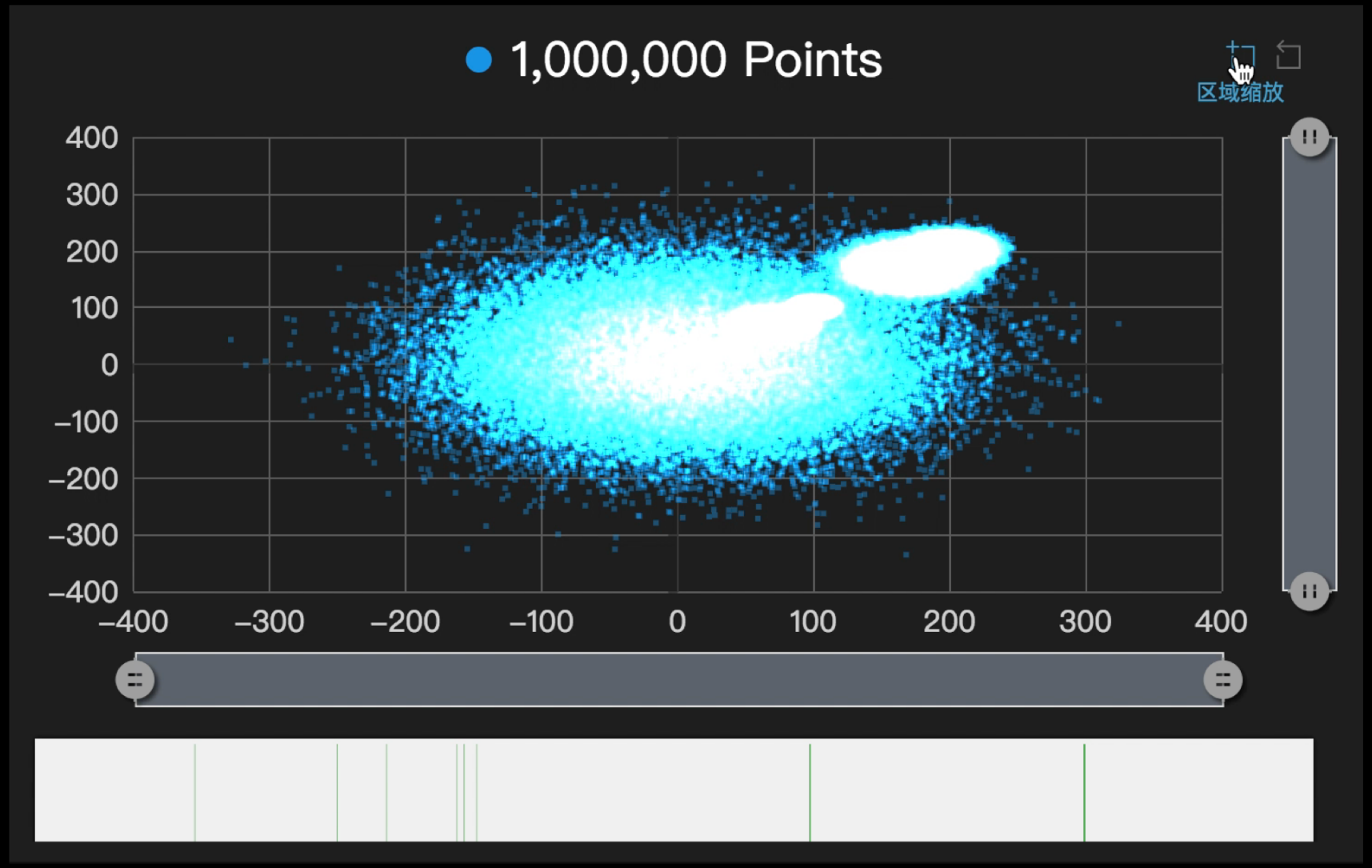
Solution 3: Perception Based Approach

- Render as much result as possible
- Respond to user interaction when still rendering

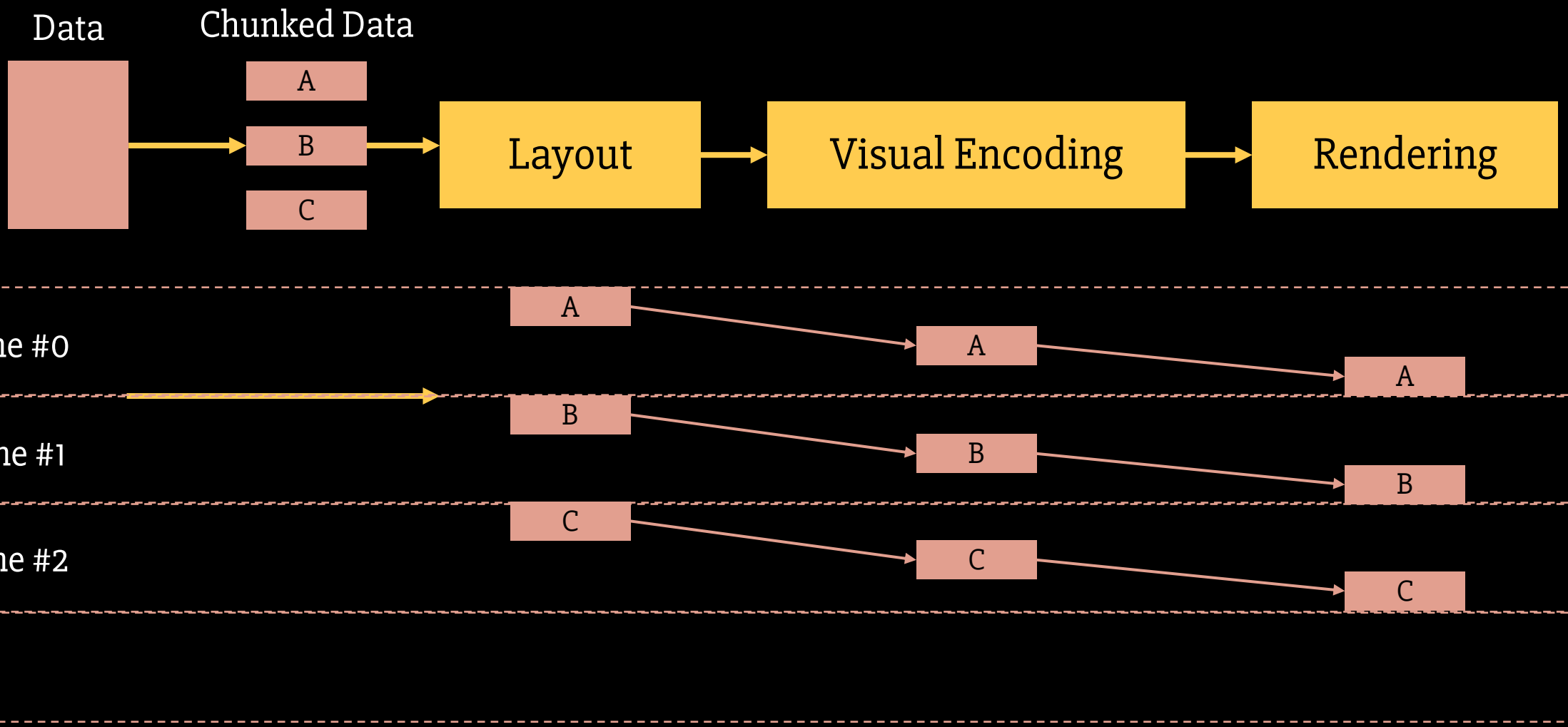
Before Progressive Rendering (ECharts v3)



After Progressive Rendering (ECharts 4)



ECharts 4 Workflow - Progressive Rendering



Problem with Canvas

Problem Fixed with WebGL

Pros & Cons
of Perception Based Approach

Pros

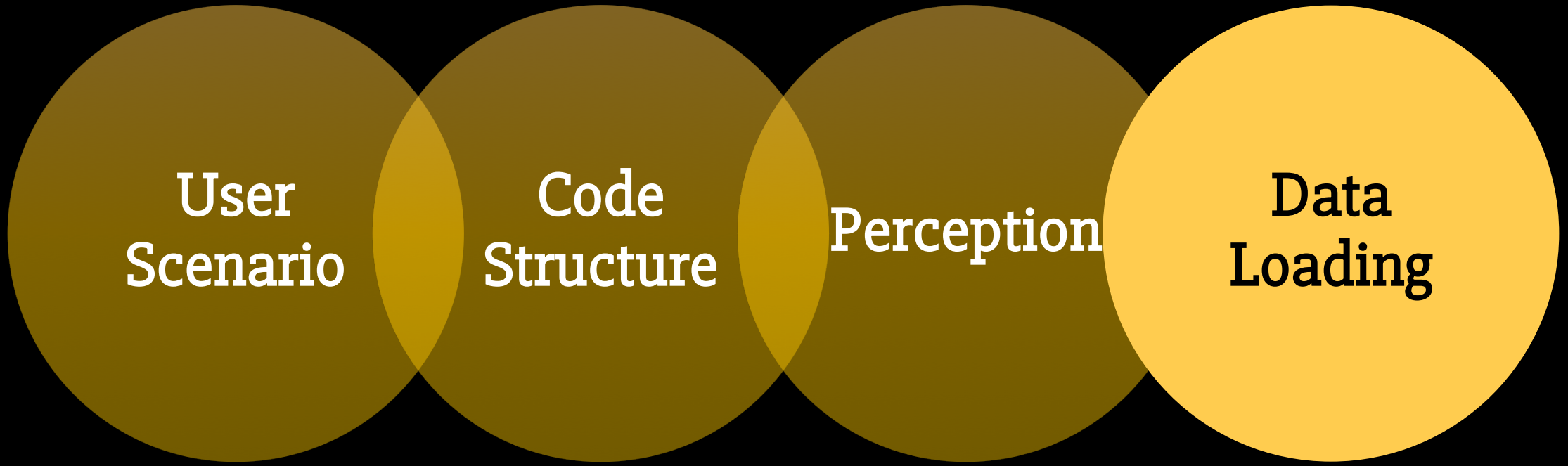
- Improves user experience
  - Instant feedback
  - Continuous progressing

Cons

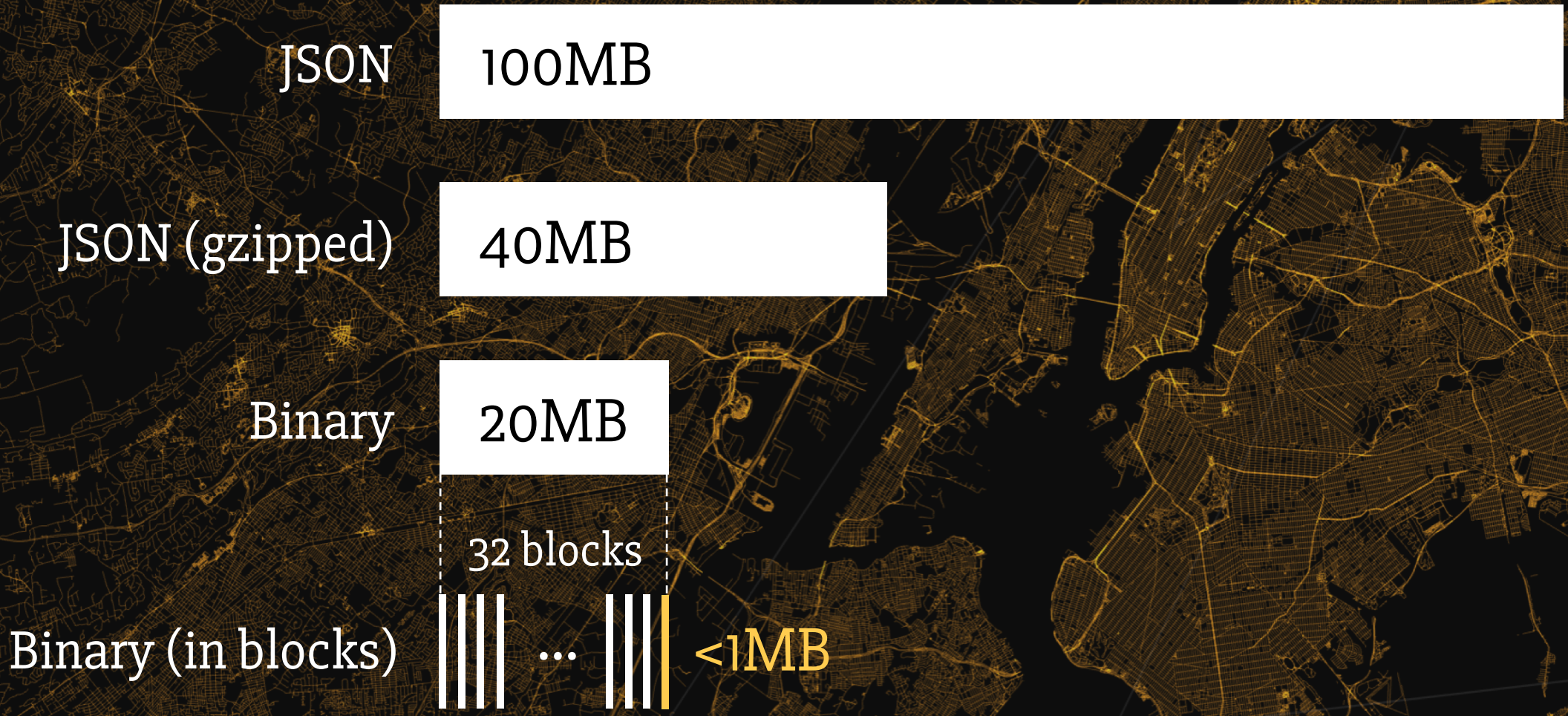
- Developing cost
  - Great change in the architecture

Use only when rendering big data

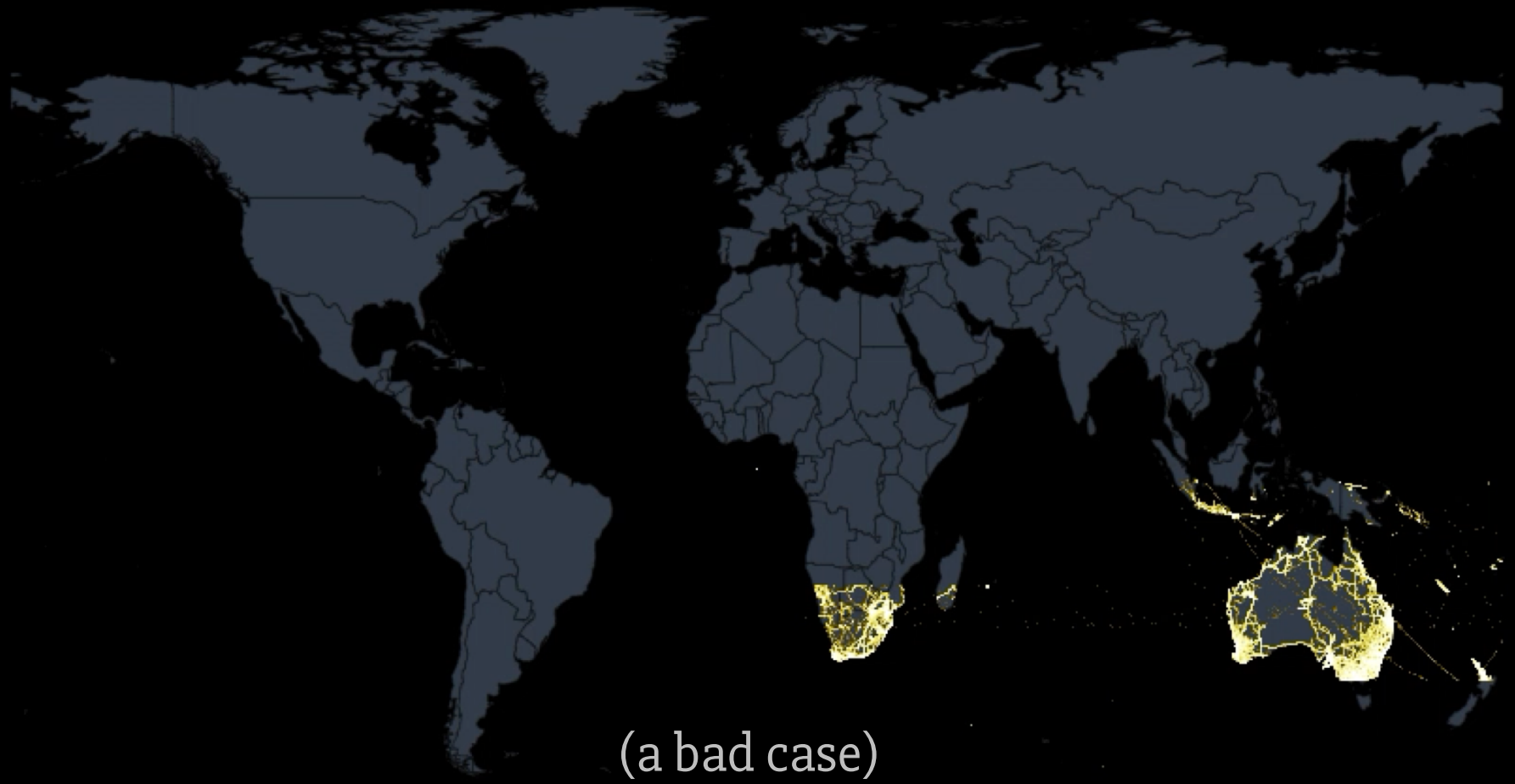
ECharts' Solution



Solution 4: Data Loading Based Approach



Split The Data Randomly



(a bad case)

Pros & Cons
of Data Loading Approach

Pros

- Improves user experience

Cons

- Developing cost

Incubation under ASF

Challenges of ECharts

- Extensive users vs. Limited contributors
  - Users: npm weekly downloads: 130k
  - Contributors:
    - 3 committers since incubation
    - < 10 core code maintainers
- Lacking international users

What we learned from the Apache Way

TO BE MORE OPEN

- More discussion and information on the mailing list
  - dev@echarts.apache.org
- More documents of everything
- Embrace the community

THANK YOU!

Wenli Zhang

 http://zhangwenli.com

 me@zhangwenli.com

 @OviliaZhang

 @Ovilia

Icons from: https://www.flaticon.com