Writing software that keeps working

Dr. Gleb Bahmutov PhD

gleb@kensho.com
glebbahmutov.com
github.com/bahmutov
@bahmutov

MIT 6.148 web competition
KENSHO

Technology that brings transparency to complex systems

Harvard Sq, WTC NYC
jobs@kensho.com
https://github.com/kensho
Dr. Gleb Bahmutov, PhD
JavaScript ninja, image processing expert, software quality fanatic

email blog github @bahmutov slides conferences videos about me

Today

I am a software engineer interested in front and back end development, especially in using JavaScript (and its flavors) across the entire stack. I work at Kensho, making the world a better place. I try to blog on topics related to software craft (more than 300 blog posts!). Here is what people say about my blog posts and presentations.

201 OSS projects

In my spare time I built some cool stuff; see the links below. I try to keep modules small because I agree with Sindre Sorhus.

>> draw-cycle is a simple Cycle.js application visualized: streams, events, DOM.
>> bottle-service is a tiny library to enable any web application to become a self-rewriting app instantly loaded from ServiceWorker cache. See an example application demo at instant-vmotodom and read the blog

glebbahmutov.com glebbahmutov.com/blog
Overview

- Developer value
- Modular software
- Testing JavaScript code
- Unsolicited advice

@bahmutov
Developer Value

At large successful companies (Facebook, Google, MathWorks) each developer brings > $1,000,000 in revenue per year.
# Developer Value

<table>
<thead>
<tr>
<th>year</th>
<th>new profit per year</th>
<th>total profit per year</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$100k</td>
<td>$100k</td>
</tr>
<tr>
<td>2</td>
<td>$100k</td>
<td>$200k</td>
</tr>
<tr>
<td>3</td>
<td>$100k</td>
<td>$300k</td>
</tr>
<tr>
<td>...</td>
<td>$100k</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>$100k</td>
<td>$1mil</td>
</tr>
</tbody>
</table>

[@bahmutov](https://glebbahmutov.com/blog/developer-value/)
## Developer Value

<table>
<thead>
<tr>
<th>year</th>
<th>new profit per year</th>
<th>total profit per year</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$100k</td>
<td>$100k</td>
</tr>
<tr>
<td>2</td>
<td>$100k</td>
<td>$200k</td>
</tr>
<tr>
<td>3</td>
<td>$100k - $100</td>
<td>$200k</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>$100k - $100</td>
<td>$200k</td>
</tr>
</tbody>
</table>

 ربما يكون من الأفضل إعادة كتابة البرمجيات من السنة الأولى.

@bahmutov

https://glebbahmutov.com/blog/developer-value/
Developer Value

"Your value keeps increasing as long as the software you wrote keeps working."

https://glebbahmutov.com/blog/developer-value/
Developer Value

How much of what you wrote today (MIT 6.148) will survive in 1/5/10 years?

Help needed: commit-survivor

@bahmutov
Hard to keep the entire car running for a long time.
Simple to reuse parts
Build complex software from simple, dependable parts

- libraries over frameworks
- simplicity over features
- functional programming over OO

@bahmutov
Reusable software

Module created for and reused by other systems gets the "natural selection" pressure, making it stronger and "antifragile"

https://glebbahmutov.com/blog/review-of-antifragile-by-nassim-nicholas-taleb/
"Put it on GitHub and reuse yourself. Tell others.

Kent C. Dodds

How to write OSS library
http://slides.com/kentcdodds/write-oss#/

The First Pull Request
http://slides.com/kentcdodds/1st-pr#/

@bahmutov
Reusable software has a public API

```
GET /users

module.exports =
  function add(a, b) {...}
```
JavaScript: a package

```
/tmp/my-code $ npm init --yes
Wrote to /tmp/my-code/package.json:
{
    "name": "my-code",
    "version": "1.0.0",
    "description": "",
    "main": "index.js",
    "scripts": {
        "test": "echo \"no tests\" && exit 1"
    },
    "keywords": [],
    "author": "",
    "license": "ISC"
}
```

node package manager (npm)

https://docs.npmjs.com/cli/init
Install dependencies

```
/tmp/my-code $ npm install --save express
/tmp/my-code $ npm i -S debug
/tmp/my-code $ npm install --save-dev qunit
/tmp/my-code $ npm i -D mocha
{
    "name": "my-code",
    "dependencies": {
        "express": "4.0.0",
        "debug": "2.6.0"
    },
    "devDependencies": {
        "qunit": "1.0.2",
        "mocha": "3.2.0"
    }
}
```

`npm help install`
What does package do?

```
/tmp/my-code $ npm info express
/tmp/my-code $ npm home express
```

![Express.js](https://expressjs.com)

**Express**

Fast, unopinionated, minimalist web framework for Node.js

```
$ npm install express --save
```

<table>
<thead>
<tr>
<th>Web Applications</th>
<th>APIs</th>
<th>Performance</th>
<th>Frameworks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Express is a minimal and flexible Node.js web application framework that provides a robust set of features for web and mobile applications.</td>
<td>With a myriad of HTTP utility methods and middleware at your disposal, creating a robust API is quick and easy.</td>
<td>Express provides a thin layer of fundamental web application features, without obscuring Node.js features that you know and love.</td>
<td>Many popular frameworks are based on Express.</td>
</tr>
</tbody>
</table>


@bahmutov

```
npm help info
npm help home
```
Example package

https://github.com/bahmutov/mit-6.148

git clone git@github.com:bahmutov/mit-6.148.git
cd mit-6.148
npm install
npm run lint
npm test

@bahmutov
NPM init (simple)

$ npm init --yes
Wrote to /tmp/my-code/package.json:

Yeoman generator

$ npm install -g yo generator-node-bahmutov
yo node-bahmutov

linter, unit testing, all necessary fields, Docker file, security, publishing, etc.

https://github.com/bahmutov/generator-node-bahmutov
API semantic versioning

1.0.0

major.minor.patch

Widely used in JavaScript world
"semver: how does API change between releases?

- **major**: API has changed
- **minor**: new API method has been added
- **patch**: a bug has been fixed, no API changes
Next version problem

current version: **1.0.0**

commits:
- add a button
- fix title
- remove notification email

"What is the new version?"
Hard to determine "meaning" of each commit

Humans should provide the meaning

```
$ git commit -m "feat(button): add login button"
```

commit type: validate-commit-msg

@bahmutov
current version: 1.0.0

- feat(button): add a button
- fix(title): new title text
- major(email): remove notification
current version: 1.0.0 (major.minor.patch)

- **feat(...): ...**  increments minor
- **fix(...): ...**  increments patch
- **major(...): ...**  increments major

next version: 2.0.0
Automate each release!

- feat(...): ...
- fix(...): ...
- major(...): ...

1.0.0

npm install --global semantic-release-cli
semantic-release-cli setup

1.1.0

1.1.1

2.0.0

https://github.com/semantic-release/semantic-release
https://juristr.com/blog/2015/10/release-like-a-pro/
Relying on versions

```
{
  "dependencies": {
    "foo": 1.0.0
  }
}
```

"foo" has versions 1.0.1, 1.1.0, 1.2.0 and 2.0.0 available

Can I upgrade to new version of "foo"?

@bahmutov
Theory

"foo" has versions 1.0.1, 1.1.0, 1.2.0 and 2.0.0 available
"foo" has versions 1.0.1, 1.1.0, 1.2.0 and 2.0.0 available

No one knows for sure!
Upgrade automation

```plaintext
~/git/chdir-promise on master
$ next-update --tldr
> tests are passing at the start

<table>
<thead>
<tr>
<th>package</th>
<th>current</th>
<th>available</th>
<th>success %</th>
<th>successful updates</th>
<th>failed updates</th>
</tr>
</thead>
<tbody>
<tr>
<td>check-more-types</td>
<td>1.1.1</td>
<td>1.7.3</td>
<td>74%</td>
<td>23</td>
<td>8</td>
</tr>
<tr>
<td>check-types</td>
<td>1.4.0</td>
<td>3.3.0</td>
<td>0%</td>
<td>0</td>
<td>26</td>
</tr>
<tr>
<td>lazy-ass</td>
<td>0.5.3</td>
<td>0.5.8</td>
<td>100%</td>
<td>20</td>
<td>0</td>
</tr>
<tr>
<td>q</td>
<td>1.1.2</td>
<td>1.4.1</td>
<td>100%</td>
<td>15</td>
<td>0</td>
</tr>
<tr>
<td>spots</td>
<td>0.3.0</td>
<td>0.5.0</td>
<td>100%</td>
<td>14</td>
<td>0</td>
</tr>
<tr>
<td>pre-git</td>
<td>0.1.1</td>
<td>0.6.0</td>
<td>100%</td>
<td>24</td>
<td>0</td>
</tr>
</tbody>
</table>

update stats from http://next-update.herokuapp.com
stats: check-more-types 1.1.1 -> 1.7.3 success probability 74% 23 success(es) 8 failure(s)
check-more-types@1.7.3 works
stats: check-types 1.4.0 -> 3.3.0 success probability 0% 0 success(es) 26 failure(s)
check-types@3.3.0 tests failed :(
stats: lazy-ass 0.5.3 -> 0.5.8 success probability 100% 20 success(es) 0 failure(s)
lazy-ass@0.5.8 works
stats: q 1.1.2 -> 1.4.1 success probability 100% 15 success(es) 0 failure(s)
q@1.4.1 works
stats: spots 0.3.0 -> 0.5.0 success probability 100% 14 success(es) 0 failure(s)
spots@0.5.0 works
stats: pre-git 0.1.1 -> 0.6.0 success probability 100% 24 success(es) 0 failure(s)
pre-git@0.6.0 works
```

https://github.com/bahmutov/next-update
Developer value

Reusable long-lasting software

Modular versioned software

Testing

@bahmutov
Beautiful tests

source: https://www.petersons.com/graduate-schools/sample-lsat-test-questions.aspx
JavaScript is unsafe at any speed

- No types
- Gotchas
- Tangled with Web standards
- 3rd party libraries

@bahmutov
Test types

end to end tests

integration tests, integration tests

unit tests, unit tests, unit tests, unit tests, unit tests, unit tests, unit tests
Test types: practice

end to end tests

integration tests, integration tests

linter, unit tests, unit tests, unit tests, crash reporting

@bahmutov
Test types: tools

- end to end tests
- cypress.io
- integration tests
- linter, unit tests
- standard, mocha, ava, jest
- sentry

@bahmutov
Linting source code

// add.js
function add(a, b) {
    return a + b
}
module.exports = addition
Setup linter

```bash
npm install --save-dev standard
```

```json
{
    "scripts": {
        "test": "echo "no tests" && exit 1",
        "lint": "standard --verbose --fix *.js",
        "pretest": "npm run lint"
    }
}
```

http://standardjs.com/

@bahmutov
Linters gonna lint

$ npm run lint
> standard --verbose --fix *.js
standard: Use JavaScript Standard Style (http://standardjs.com)
   add.js:1:10: 'add' is defined but never used. (no-unused-vars)
   add.js:4:18: 'addition' is not defined. (no-undef)

http://standardjs.com/

@bahmutov
Fixed lint errors

// add.js
function add(a, b) {
    return a + b
}

module.exports = add
Unit testing

```bash
npm install --save-dev mocha
```

```json
{
    "scripts": {
        "test": "mocha *-spec.js",
        "lint": "standard --verbose --fix *.js",
        "pretest": "npm run lint"
    }
}
```

http://mochajs.org/

@bahmutov
describe('add', function () {
  const add = require('./add')
  it('adds two numbers', () => {
    console.assert(add(2, 3) === 5)
  })
})

https://glebbahmutov.com/blog/unit-test-node-code-in-10-seconds/
Promise testing

```javascript
const five = Promise.resolve(5);
describe('five', function () {
  it('resolves to 5', () => {
    return five.then(value =>
      console.assert(value === 5)
  )
})
})
```

http://mochajs.org/#asynchronous-code
Unit test choices

test library | predicates and assertions
---|---
Mocha | Chai
Ava | lazy-ass + check-more-types
Jest |
Crashes will happen

Sentry.io crash service

```javascript
if (process.env.NODE_ENV === 'production') {
    var raven = require('raven');
    var SENTRY_DSN = 'https://<DSN>@app.getsentry.com/...';
    var client = new raven.Client(SENTRY_DSN);
    client.patchGlobal();
}
foo.bar // this Error will be reported
```

https://glebbahmutov.com/blog/know-unknown-unknowns-with-sentry/
@bahmutov https://glebbahmutov.com/blog/stand-your-own-crash-server/

npm install raven --save
src:app at Object.handle
Error: Do not know what to do for /

users  1  3 minutes ago root

Event at Dec. 26, 2013, 7:46 p.m. [35.9 KB]
ID: 14032a9da28646d3a7c3b21de2c30436

Tags
- level = error
- logger = root
- server_name = Powerline.local
- url = http://localhost:3000/

Exception (most recent call first)

Error
Do not know what to do for /

/Users/gbahmutov/git/proud-connect/src/app.js ? in Object.handle (application)

175. throw new Error('Do not know what to do for ' + req.url);

Request

@bahmutov
### Exception (most recent call first)

#### Error

Do not know what to do for /

```
/Users/gbahmutov/git/proud-connect/src/app.js in Object.handle (application)
168.    .use(connect.logger('dev'))
169.    .use(connect.static('public'))
170.    .use(connect.bodyParser())
171.    .use(connect.cookieParser())
172.    .use(connect.query())
173.    .use(function (req, res, next) {
174.      if (!req.url || req.url === '/') {
175.        throw new Error('Do not know what to do for ' + req.url);
176.        next();
177.        return;
178.      }
179.      res.writeHead(401, 'missing NPM username');
180.      res.end();
181.      return;
182.    }
```

### Request

**URL:** http://localhost:3000/

**Method:** GET
Crash early and often

“Defensive coding: checking inputs before each computation (preconditions). Sometimes checking result value before returning (postconditions).
Crash early and often

“Paranoid coding: checking inputs before each computation, as if the caller was evil and trying to break the function on purpose.”

https://glebbahmutov.com/blog/paranoid-coding/

@bahmutov
Crash early and often

const la = require('lazy-ass')
const is = require('check-more-types')
la(is.strings(names), 'expected list of names', names)
la(is.email(loginName), 'missing email')
la(is.version(tag) || check.sha(tag), 'invalid tag', tag)

https://glebbahmutov.com/blog/paranoid-coding/
Test types: tools

end to end tests

linter, unit tests, unit tests, unit tests, crash reporting

standard  mocha, ava, jest  sentry

@bahmutov
Multiple moving parts

generous source: http://www.goodwp.com/tags/mechanism/
Grab source code at

https://github.com/bahmutov/2048-kensho

Build and run local website

`npm install`
`npm run build`
`npm start`

Install and start Cypress

`npm install -g cypress-cli`
`cypress install`
`cypress open`
Login with GitHub

To get invite:
- enter email at https://www.cypress.io/#footer
- ask politely at https://gitter.im/cypress-io/cypress

@bahmutov
Added this project to Cypress

@bahmutov
describe('2048-kensho', function(){
  beforeEach(function(){
    cy.visit('http://localhost:3000')
  })
  it('has 2048 title', function(){
    cy.title().should('include', '2048')
  })
})

Mocha + Chai assertions

https://docs.cypress.io/docs/making-assertions

@bahmutov

cypress/integration/2048-kensho-spec.js
Set "baseUrl" in cypress.json

```javascript
{
    "baseUrl": "http://localhost:3000"
}

describe('2048-kensho', function(){
    beforeEach(function(){
        cy.visit('/')
    })
    it('has 2048 title', function(){
        cy.title().should('include', '2048')
    })
})

@bahmutov
```
Test any website by changing baseUrl

CY PRESS_baseUrl=https://glebbahmutov.com/2048-kensho cypress open

Every time you refresh the page, the team is reshuffled.

Created by G leb Bahmutov from the "official" github source authored by Gabriele Cirulli. Source at github
it('starts with 3 tiles', () => {
    cy.get('.tile-container')
    .find('.tile')
    .should('have.length', 3)
})

it('has header', () => {
    cy.get('h1.title').should('be.visible')
    .contains('kensho')
})


Examples: https://docs.cypress.io/docs/all-example-apps
Summary

- Start a versioned package (free)
- Lint and unit test (free)
- Use continuous integration (CI) server (free)
- Setup crash reporting (free)
- E2E test using Cypress (free)
A bit of advice

“Simplicity always wins in the long term
Last bit of advice

You must do...

Dynamic website supported by a back-end
Zeit Now, Dokku

Personalized experience based on user accounts
Auth0

Minimum security requirements fulfilled
SSL and CSP

Original design and implementation
all you

Use Git on a 6.148 Github repo
git-extras

https://glebbahmutov.com/blog/think-inside-the-box/

https://github.com/tj/git-extras

@bahmutov
Thank you and good luck

https://slides.com/bahmutov/working-software

Questions, career advice, public speaking, professional development, mentorship - ping me, will be happy to help

gleb@kensho.com
glebbahmutov.com
github.com/bahmutov
@bahmutov

MIT 6.148 web competition