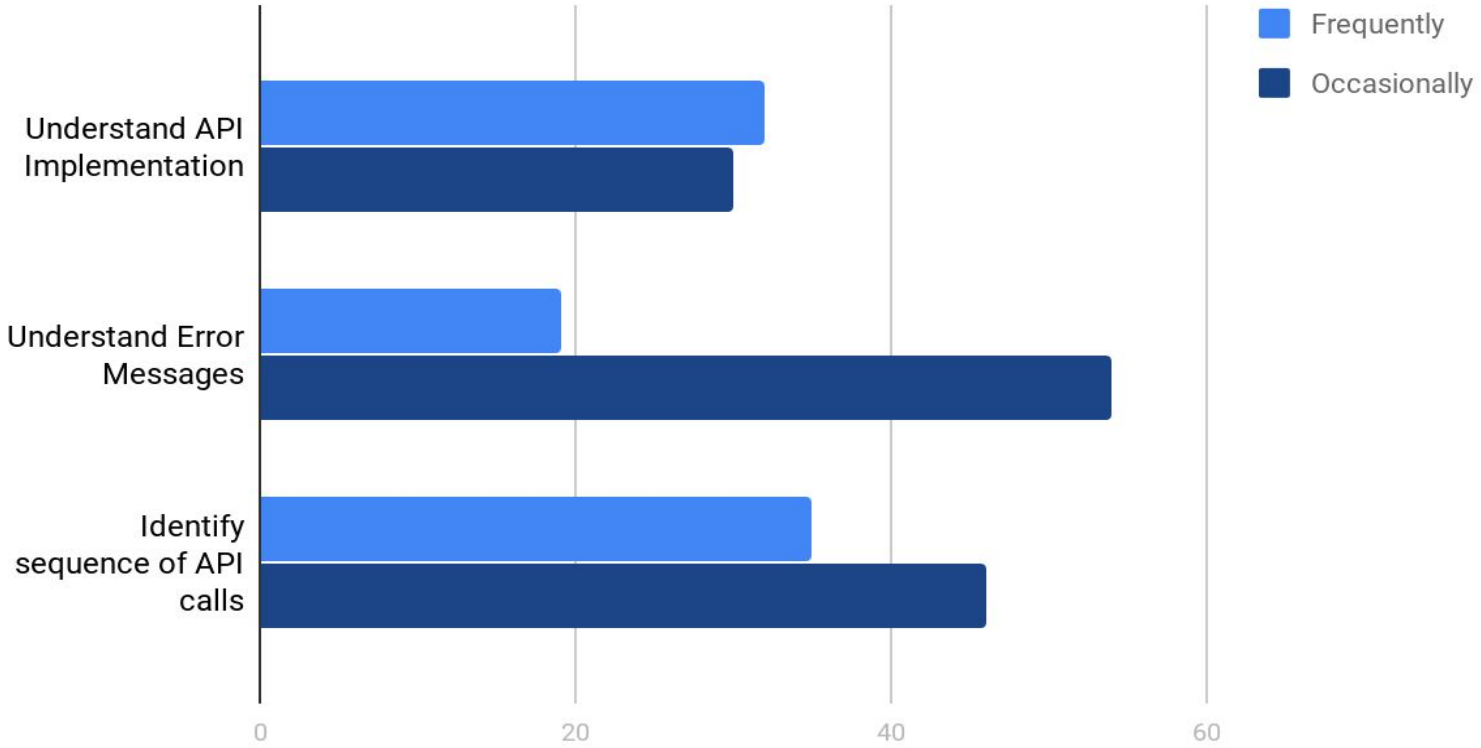
Fluent API for NodeJS Crypto

Simon Kafader

Obstacles in using crypto API



```
let algorithm = 'aes-128-cbc';
let ivLength = 16;
let keyLength = 16;
let saltLength = 64;
let iterations = 1000;

let password = 'This is a private key';
let randomIV = crypto.randomBytes(ivLength);
let salt = crypto.randomBytes(saltLength);
let key = crypto.pbkdf2Sync(password, salt, iterations,
    keyLength, 'sha512');

let cipher = crypto.createCipheriv(algorithm, key,
    randomIV);
let cipherText = cipher.update(toEncrypt, 'utf8', 'hex');
cipherText += cipher.final('hex');
```

My Goal

Provide an easier way for developers to use Crypto APIs

Background

Domain Specific Language (DSL)

Languages focused on a particular application domain

- Human-Readable, Computer-Executable
- Limited expressiveness
- Focus on particular domain

E.g. SQL, HTML

```
USE AdventureWorks2012;
GO
SELECT Name, ProductNumber, ListPrice AS Price
FROM Production.Product
WHERE ProductLine = 'R'
AND DaysToManufacture < 4
ORDER BY Name ASC;
GO
```

Fluent API/Interface

Design Pattern

Particular kind of DSL that emphasizes language-like flow

Mostly based on method chaining

JQuery

```
$('div.test')
   .on('click', handleTestClick)
   .addClass('foo');
```

My Approach

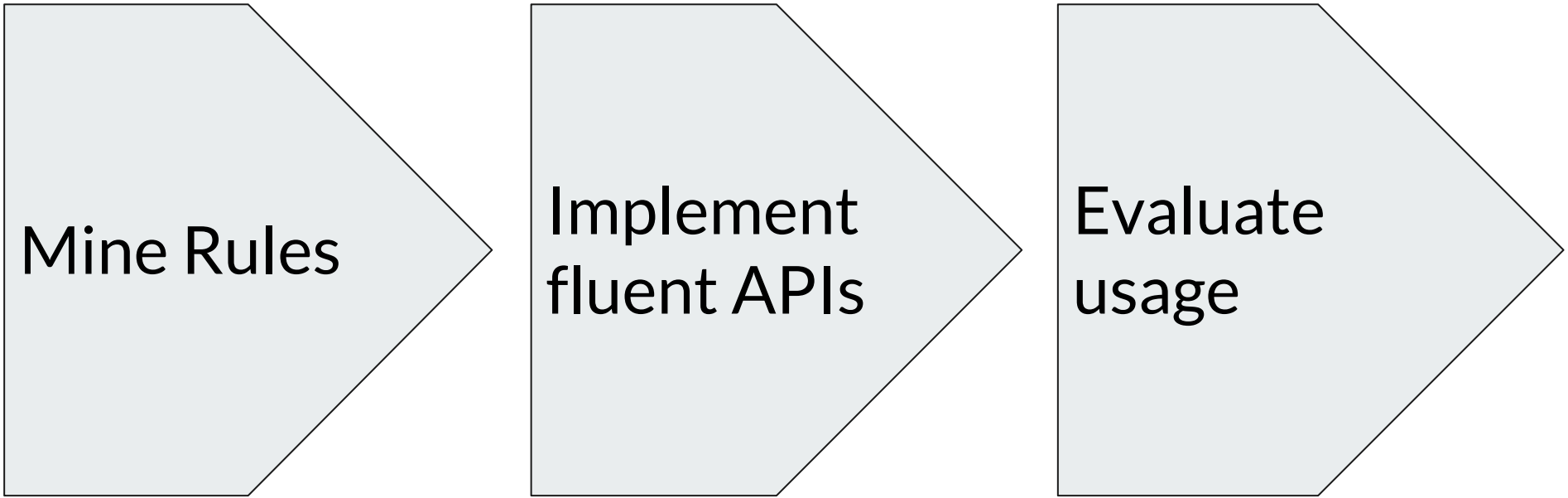
Fluent API for NodeJS Crypto APIs

Improve readability,

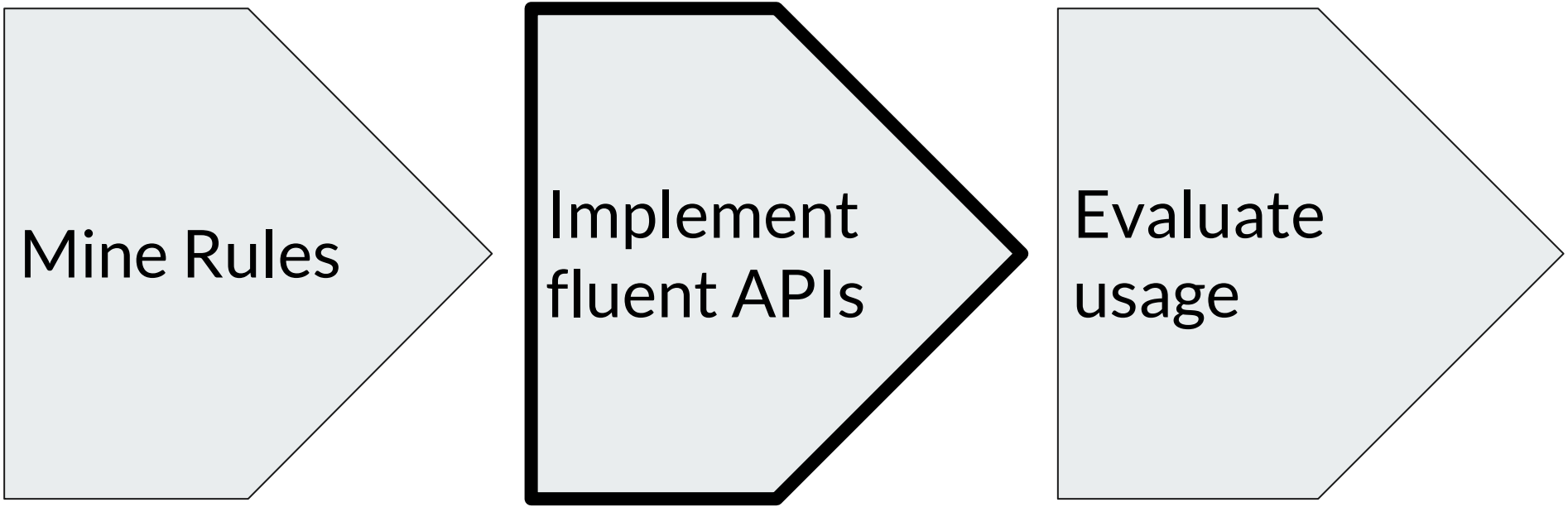
provide detailed, and

helpful error messages

The Process



The Process



```
let crypto = require('crypto');

let password = 'This is a private key';
let randomIV = crypto.randomBytes(16);
let salt = crypto.randomBytes(64);
let key = crypto.pbkdf2Sync(password, salt, 1000,
    16, 'sha512');

let cipher = crypto.createCipheriv('aes-128-cbc', key,
    randomIV);
let toEncrypt = 'I want to be encrypted';
cipher.update(toEncrypt, 'utf8', 'hex');
let cipherText = cipher.final('hex');

let decipher = crypto.createDecipheriv('aes-128-cbc', key, randomIV);
decipher.update(cipherText, 'hex', 'utf8');
let decrytped = decipher.final('utf8');
```

```
let iv = fluentCrypto.generateRandomIVFor('aes-128-cbc');

let secretKey = fluentCrypto.generateSymmetricKey()
    .for('aes-128-cbc')
    .generate();

let toEncrypt = 'I want to be encrypted';

let cipherText = fluentCrypto.createCipher(iv, secretKey, 'aes-128-cbc')
    .data(toEncrypt)
    .cipher();

console.log(cipherText);

let decipheredText = fluentCrypto.createDecipher(iv, secretKey, 'aes-128-cbc')
    .data(cipherText)
    .decipher();
```