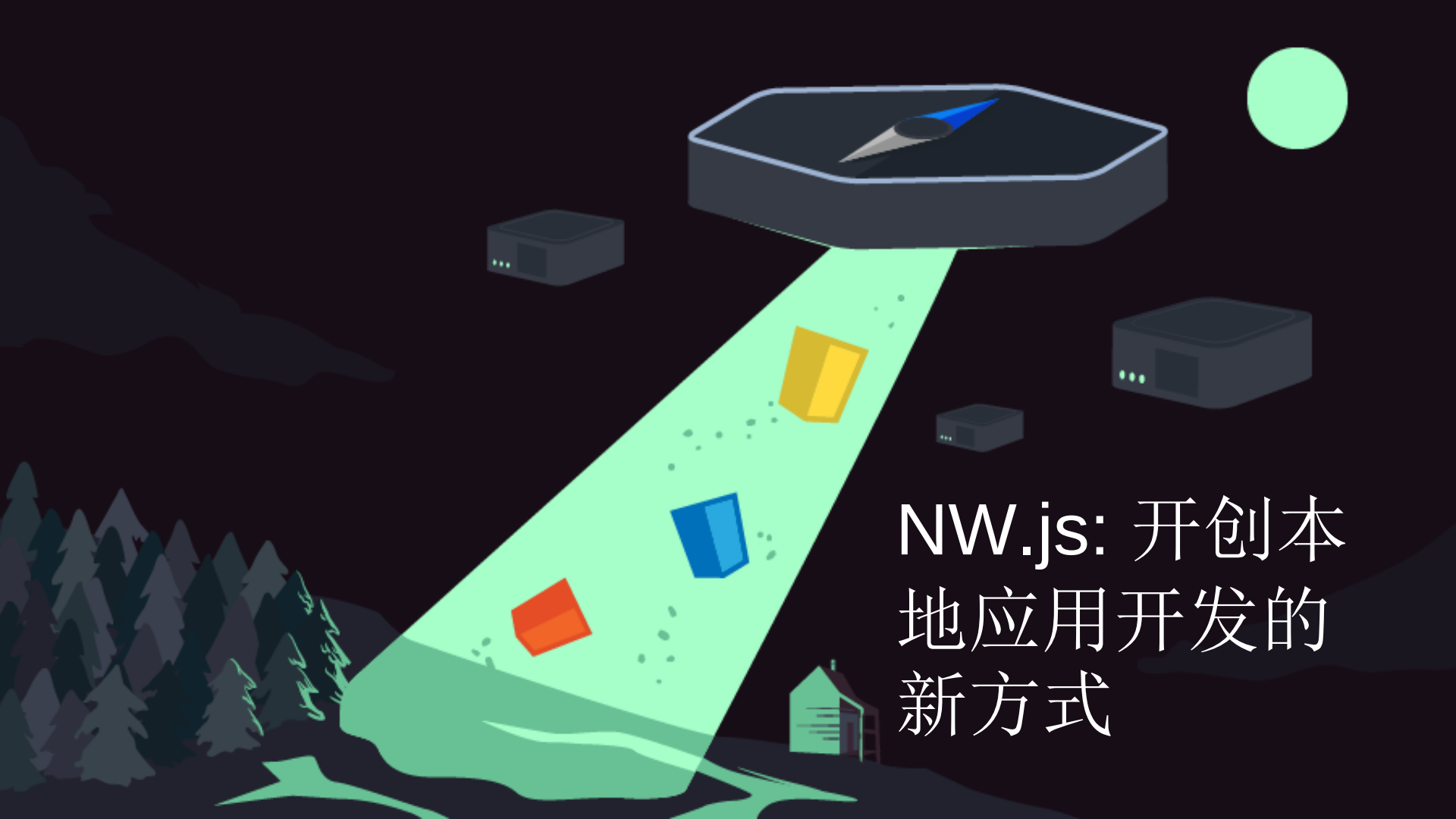

NW.js: 开创本地应用开发的新方式

NW.js 简介

在DOM中直接调用Node.js

```
<!DOCTYPE html>
<title>Call all Node modules directly from DOM</title>
<script>
  var fs = require('fs');
  fs.open(....
</script>
```

发布NW.js应用

```
app/
├── nw.exe ...
├── js/
│   └── main.js
├── assets/
│   └── icon.png
├── styles/
│   └── common.css
├── views/
│   └── main.html
└── package.json
```

package.json:

```
{
  "name":"nw.js-example",
  "main":"views/main.html",
  "window":{
    "min_width":400,
    "min_height":400,
    "icon":"assets/icon.png"
  }
}
```

NW.js 简介(2): NW.js = Chromium + Node.js

- 本地GUI API
- 本地安全模型
- 支持chrome.* API和Chrome App
- 源代码保护

- Content Verification: 加载签名后的应用文件
- 编写多线程JS代码：在Web Worker中可使用Node

安全模型：桌面应用程序 vs Web

- 需要安装；安装代表被信任
- 可以在系统中做几乎任何事情
- 通过受信任的渠道分发

- 从互联网上直接下载，在浏览器中执行
- 默认不受信任
- 做特殊操作时需要用户授权

NW的本地安全模型

- 可调用本地API (Node.js, nw.*, chrome.*)
- 跨域访问
- JS调用用户交互类操作
  - 给File Input赋值
  - 模拟用户操作(触发File Input的click事件)
  - 读写剪贴版
  - 弹出窗口
  - 播放媒体标签
  - ……

```
var f = new File('/path/to/file', 'name');
var files = new FileList();
files.append(f);
document.getElementById('input0').files = files;
```

NW想让JavaScript开发者能做到和C/C++开发者同样的事

Any application that can be written in JavaScript, will eventually be written in JavaScript. - Jeff Atwood (2007)

NW想让JavaScript开发者能做到和C/C++开发者同样的事

- 通过Node使用本地API
- 使用本地安全模型，不受Web安全模型限制
- 源代码保护
- 编写多线程应用

NW源代码保护

- 将JavaScript源文件编译成CPU执行的二进制代码
- nwjc工具
- 在NW应用中加载二进制代码
- Function.prototype.toString()
- 性能考虑

在NW中编写多线程应用 - Web Worker

启动Web Worker:

```
var myWorker = new Worker("worker.js");
```

Worker.js:

```
onmessage = function(e) {
  console.log('Message received from main script');
  var workerResult = 'Result: ' + (e.data[0] * e.data[1]);
  console.log('Posting message back to main script');
  postMessage(workerResult);
}
```

main.js:

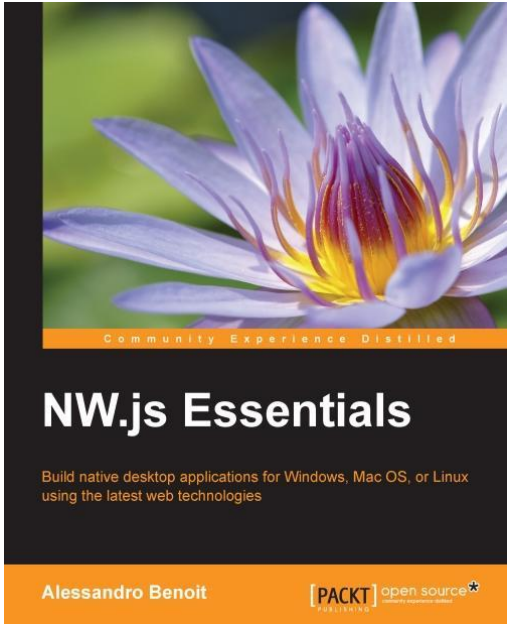
```
myWorker.onmessage = function(e) {
  result.textContent = e.data;
  console.log('Message received from worker');
}
```

使用Web技术编写桌面应用的优势

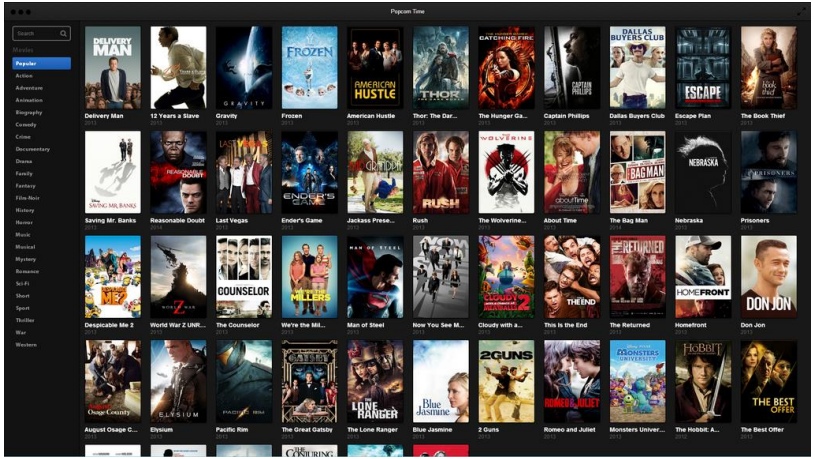
- 内容和代码重用
- 技术流行、容易找到开发者
- 只需兼容一种浏览器，并使用JavaScript和浏览器提供的新功能
  - NW 0.24 beta - Chromium 60: Paint Timing API, CSS font-display, Credential Management API improvements
  - NW 0.23 - Chromium 59: Headless, native notification on macOS, image capture API
  - NW 0.22 - Chromium 58: IndexDB 2.0; iframe navigation improvement
  - NW 0.21 - Chromium 57: CSS Grid Layout; WebAssembly
  - NW 0.20 - Chromium 56: CSS position sticky; Web Bluetooth;
  - NW 0.19 - Chromium 55: async/await; 35% 内存节省

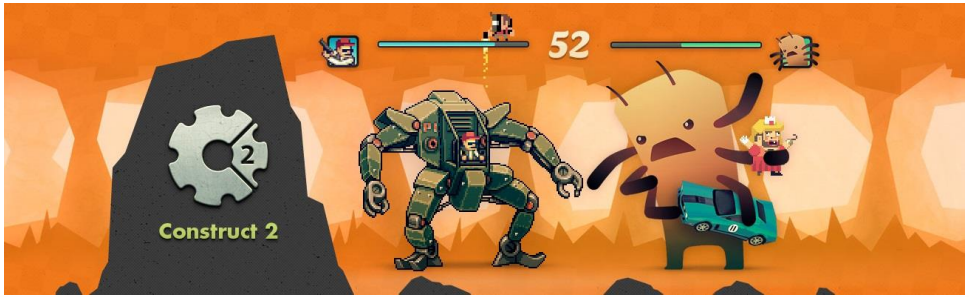
NW.js开源项目

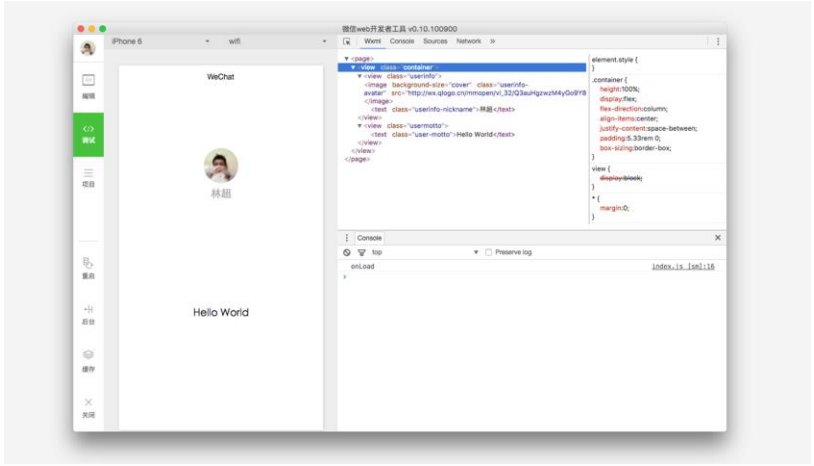
- 2011年: node-webkit创立并开源
- 2013年度中国优秀开源项目
- 2015年1月: 更名为NW.js
  - 2013年: Google发布Blink (WebKit fork)
  - 2014年底: IO.js (Node.js fork)
  - 2015年中: IO.js 和 Node.js 合并
- 2016年3月：发布0.13
  - 优化的架构 - 支持更多浏览器特性
- 在Chromium/Node升级新版本1-2天内发布对应版本，包括安全更新

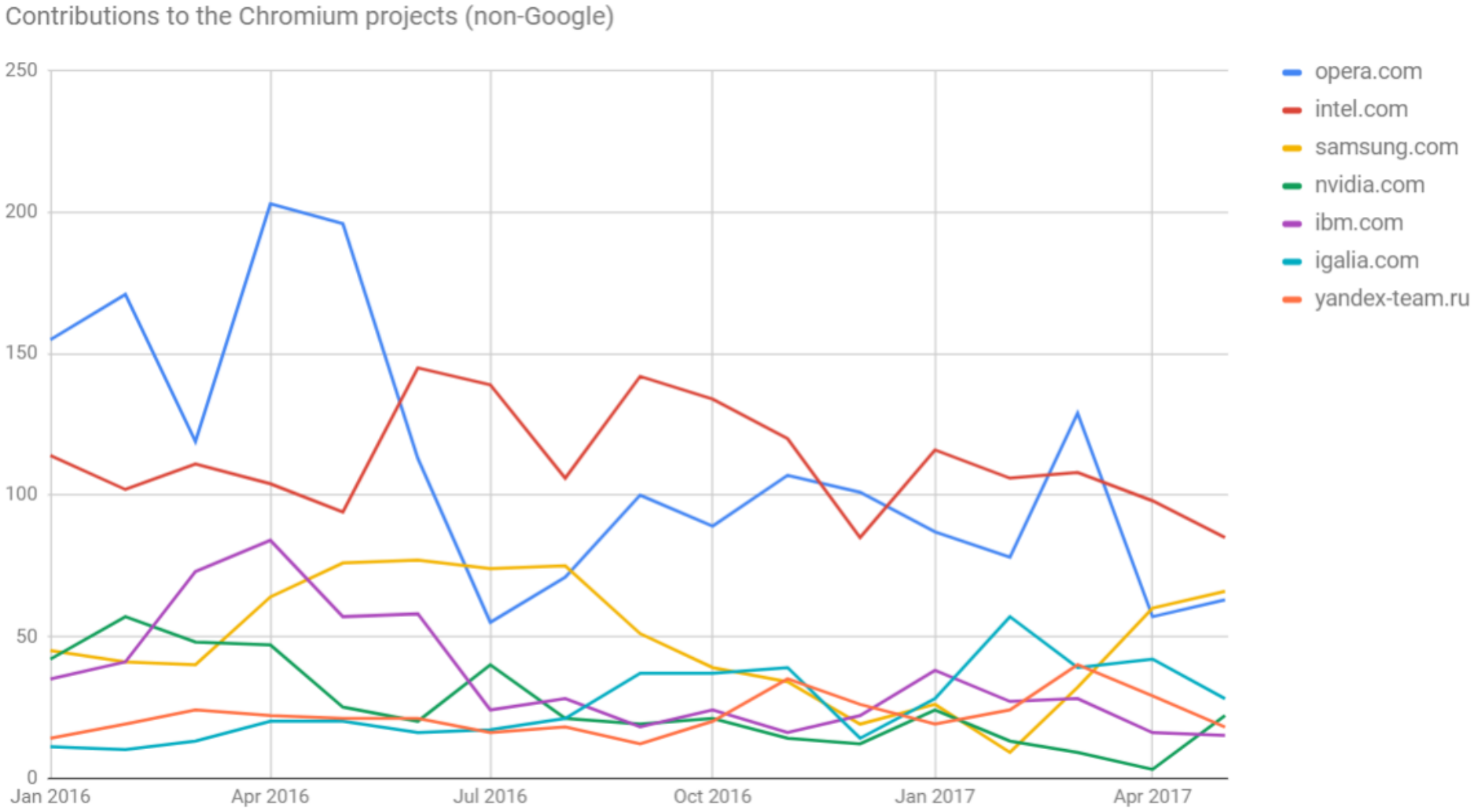

Community Experience Distilled
NW.js Essentials
Build native desktop applications for Windows, Mac OS, or Linux using the latest web technologies
Alessandro Benoit
[PACKT] open source


CSS 3
HTML 5
node-webkit
node
JS
Windows Desktop App Creation with node-webkit
From idea to .exe, without Geek-speak or Techno-jargon
by Jonathan Dodd


Popcorn Time


52
Construct 2


WeChat
Hello World


Contributions to the Chromium projects (non-Google)
250
200
150
100
50
0
Jan 2016
Apr 2016
Jul 2016
Oct 2016
Jan 2017
Apr 2017
opera.com
intel.com
samsung.com
nvidia.com
ibm.com
igalia.com
yandex-team.ru

Inference Library for JavaScript*

https://github.com/TianyouLi/Inference-Library-for-JavaScript

- JavaScript Language Interface
- Clean API for Inference/Scoring
- Extensible Architecture
- Tuned Performance for IA
- Support TensorFlow, Caffe, and MXNet; other frameworks integration is under development

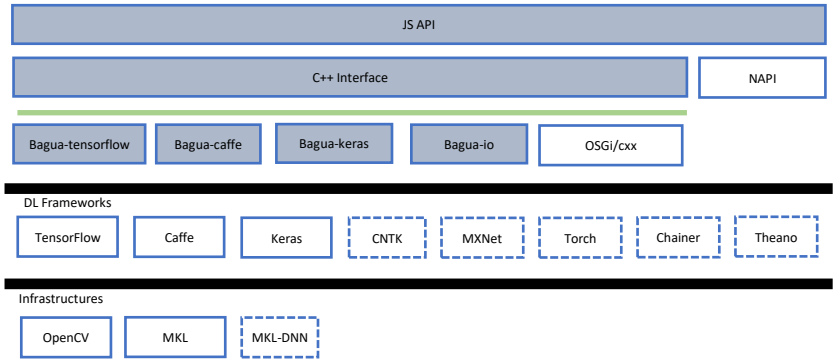
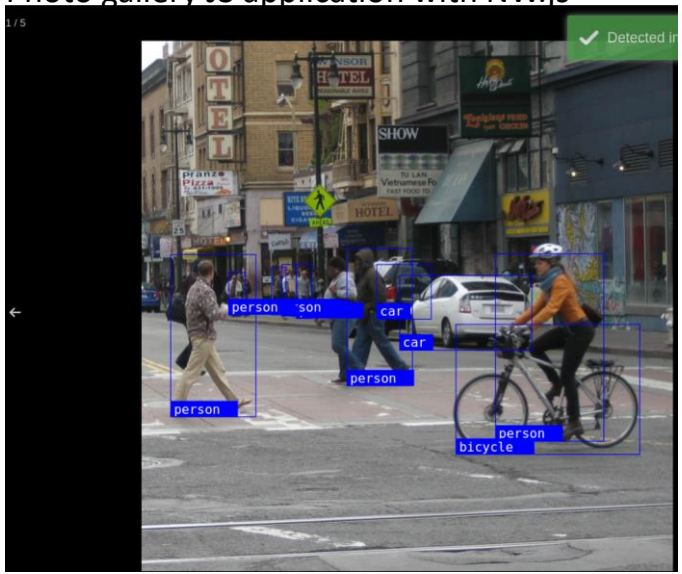
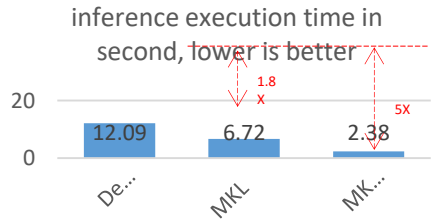


Photo gallery JS application with NW.js





Machine : N3700 (ATOM)
Caffe model: COCO 300x300, 80 categories

https://nwjs.io