FACTORY

JAVASCRIPT OD `null`-E

Domagoj Štrekelj

16. svibnja 2018.

Osijek

28

EMPLOYEES

2

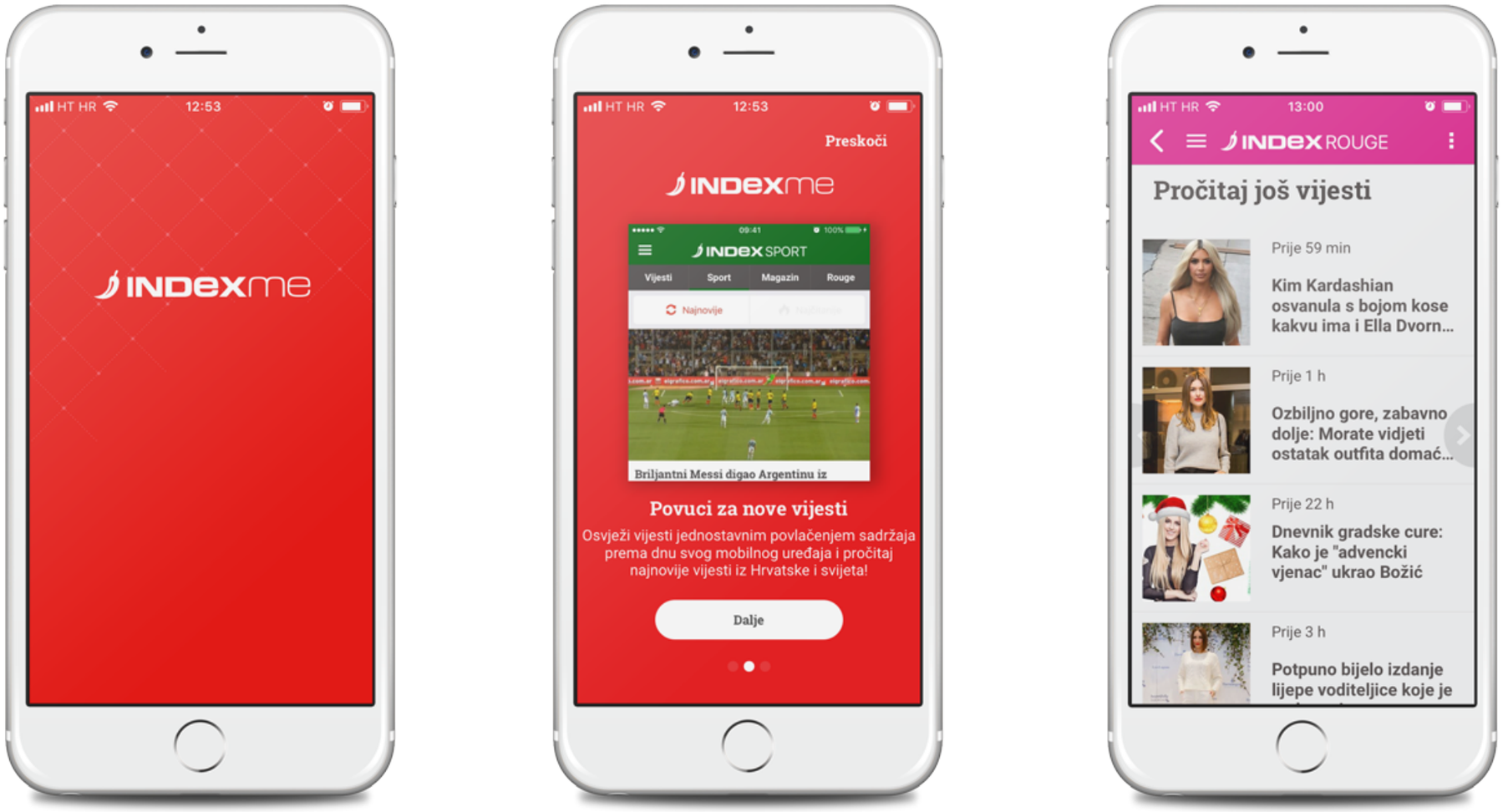
OFFICES

100+

CLIENTS

Factory u brojkama

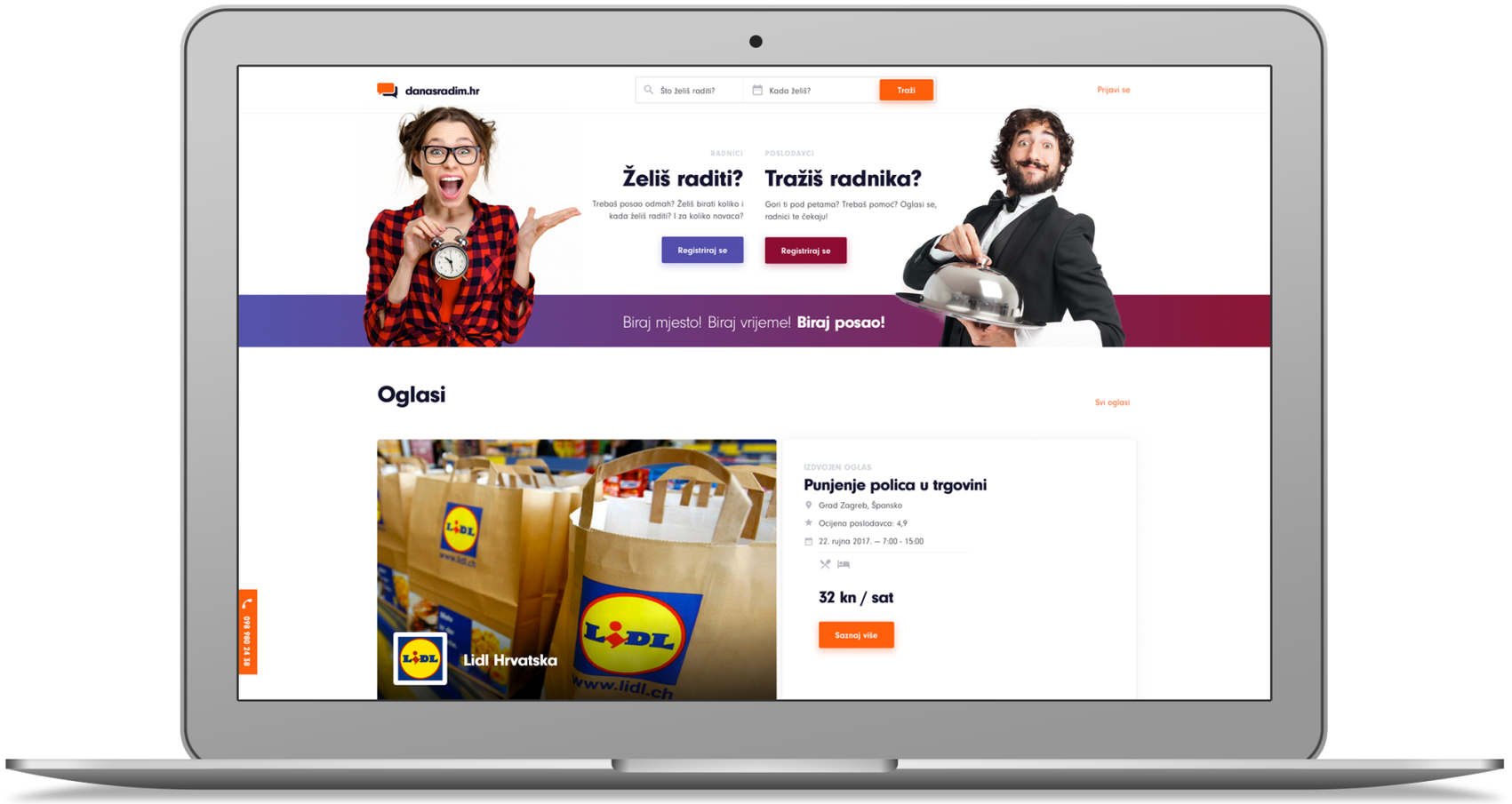
INDEX.ME



2017

Razvili smo mobilnu aplikaciju za najpoznatiji news portal u Hrvatskoj!

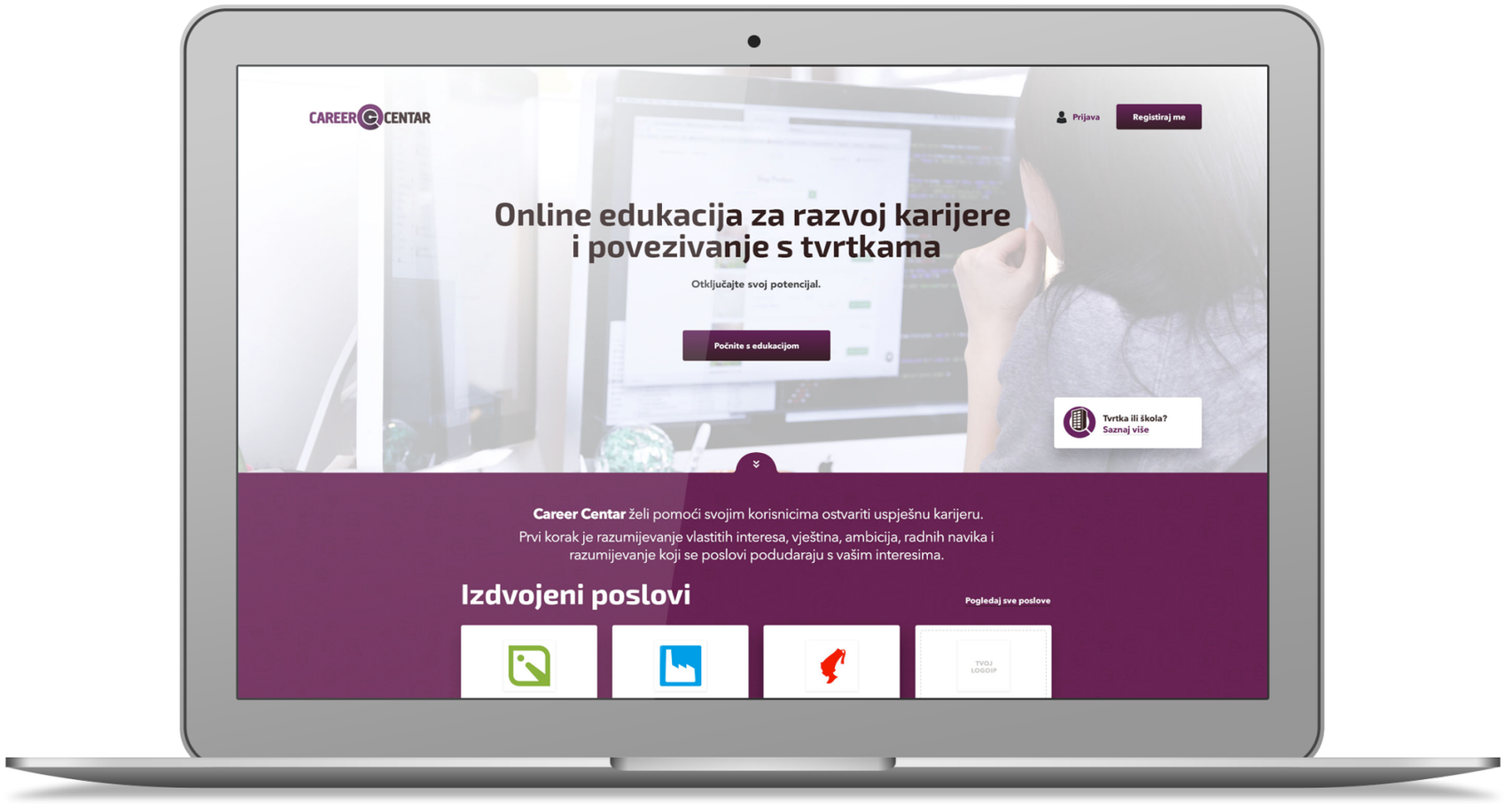
DANASRADIM.HR



Portal za pronalazak posla ili radnika na par sati, nekoliko dana, ili cijelu sezonu!

2018

CAREERCENTAR.HR



Online edukacija za razvoj karijere i povezivanje s tvrtkama.

2017

TKO SAM JA?

Domagoj Štrekelj

domagoj.strekelj@gmail.com

https://github.com/dstrekelj

@dstrekelj

domagoj.strekelj

SADRŽAJ

„It was the best of times,
it was the worst of times...”

Charles Dickens, književnik

ZAŠTO JE JAVASCRIPT?

- Priča započinje u svibnju **1995. godine**
- **Netscape Communicator** i **Mosaic** natječu se za prevlast na tržištu web preglednika
- **Marc Andreessen**, osnivač Netscapea, maša o **dinamičnom webu**
- Netscape zapošljava **Brendana Eicha** s ciljem izrade **„Schemea za preglednik”**
- Nastaje **Mocha – jednostavan, dinamičan, i pristupačan skriptni jezik za web**

Netscape

Mosaic

Marc Andreessen

Brendan Eich

ZAŠTO JE JAVASCRIPT?

- U jeku popularnosti Jave, **Sun Microsystems lobira za podršku Jave u Netscape pregledniku**
- Rukovodstvo Netscapea lobira za postojeće jezike: Python, Tcl, Scheme, ...
- **Zašto web uopće treba dva programska jezika: Javu i skriptni jezik?**

- „We aimed to provide a 'glue language' for the web designers and part-time programmers who were building web content from components [...]. We saw Java as the 'component language' used by higher-priced programmers, where the glue programmers – the Web page designers – would assemble components [...] using a scripting language."

ZAŠTO JE JAVASCRIPT?

- Rukovodstvo Netscape-a odlučuje se za implementaciju skriptnog jezika
- Popularnost Jave i pritisak Sun Microsystemsa uvjetuju da **jezik sintaksom mora biti sličan Javi**
- Pod pritiskom Netscapea, **Brendan Eich izrađuje prototip jezika u 10 dana**
- **Mocha je rođena** - jezik se integrira u Netscape Comunicator u svibnju 1995.
- Marketing odjel Netscapea mijenja naziv jezika u **LiveScript** u narednim verzijama zbog postojećih usluga sa „Live" nazivom
- **Početkom prosinca 1995. odlučuju se za ime JavaScript zbog velike popularnosti Jave**

„You'll understand JavaScript when you've forgotten what you understood before."

Angus Croll, Twitter Engineer

(parafraza izreke književnika Itala Calvina)

ŠTO JE JAVASCRIPT?

- **JavaScript je programski jezik** sa sljedećim karakteristikama:
  - Dinamičnost
  - Podrška za više programskih paradigmi
  - Standardiziranost
  - Izvorni kod se interpretira
  - Ključan dio web platforme

ŠTO JE JAVASCRIPT?

- **JavaScript uživa široku primjenu:**
  - Izrada korisničkih sučelja na klijentskoj strani (front-end) (nativno)
  - Rad s bazama podataka i posluživanje datoteka (back-end) (**Node.js**)
  - Izrada desktop (**Electron**) i mobilnih aplikacija (**Ionic**)

ŠTO JE JAVASCRIPT?

- **Program** je skup naredbi koje računalo upućuju kako riješiti zadatak
- Pravila pisanja valjanih naredbi diktira **sintaksa programskog jezika (syntax)**

```
1  function sumOdd(end) {
2      let sum = 0;
3      do {
4          if (end % 2 === 0) continue;
5          sum += end;
6      } while(end-- > 0);
7      return sum;
8  }
9
10 console.log(sumOdd(4));
11
```

ŠTO JE JAVASCRIPT?

- **Naredba (statement)** je skup riječi, brojeva, i operatora koji djeluje s određenom namjenom
- Naredbu čine jedan ili više **izraza (expression)**, odnosno varijabli ili vrijednosti povezanih operatorima
- Skup naredbi čini **blok (block)**
- **Popisi naredbi prevode se pomoću interpretera ili compilera u strojni jezik,** kojeg potom izvršava računalo

```
1  function sumOdd(end) {
2      let sum = 0;
3      do {
4          if (end % 2 === 0) continue;
5          sum += end;
6      } while(end-- > 0);
7      return sum;
8  }
9
10 console.log(sumOdd(4));
11
```

ŠTO JE JAVASCRIPT?

- **Varijable** su spremnici vrijednosti
- **Operatori** provode operacije nad varijablama i vrijednostima
- **Tipovi** predstavljaju različite vrste vrijednosti
- **Uvjetno grananje** utječe na tijek izvođenja programa
- **Petlje** izvršavaju naredbu ili blok naredbi dok je zadani uvjet ispunjen
- **Funkcije** grupiraju naredbe u zasebne, odvojene cjeline

```
1  function sumOdd(end) {
2      let sum = 0;
3      do {
4          if (end % 2 === 0) continue;
5          sum += end;
6      } while(end-- > 0);
7      return sum;
8  }
9
10 console.log(sumOdd(4));
11
```

ŠTO JE JAVASCRIPT?

- **JavaScript veže podatkovni tip uz vrijednost (value type)**, ne uz varijablu
  - **Primitivi (primitive types)**: `string`, `number`, `boolean`, `null`, `undefined`, `symbol`
  - **Složeni (compound types)**: `object`
- **Primitivi su nepromjenjivi (immutable)**, prosljeđuju se u varijable kao sama vrijednost primitiva
- **Složene tipove se može mutirati**, prosljeđuju se u varijable kao reference

ŠTO JE JAVASCRIPT?

- **Objekti** sadrže svojstva (**property**) koja mogu sadržavati vrijednosti bilo kojeg tipa
  - Svojstvima se pristupa preko **dot notacije** (`foo.bar`) ili **bracket notacije** (`foo['bar']`)
- **Niz (array) i funkcija (function) su podvrste objekta**
  - Niz je objekt koji može sadržavati vrijednosti bilo kojeg tipa pod numeričkim indeksima
  - Funkcija je objekt koji se može pozvati
- **Informacija o podatkovnom tipu** dobiva se uporabom `typeof` operatora (npr. `typeof 42`)

ŠTO JE JAVASCRIPT?

- Operator `typeof` vraća **jednu od sedam konstanti**: „string”, „number”, „boolean”, „symbol”, „object”, „function”, „undefined”
- `typeof null === 'object'` **je iskonski bug** na kojem se danas temelji moderan web
- Vrijednosti su se pohranjivale u 32-bitne strukture, zajedno s tipom (veličine do 3 bita)
- Tip „object” predstavljen je sa 000
- `null` se pohranjivao kao 0x00, zbog čega dolazi do krivog čitanja

Vrijednost	Tip	typeof
‘foobar’	string	string
42	number	number
true	boolean	boolean
Symbol(1)	symbol	symbol
{ … }	object	object
function() {}	object	function
[…]	object	object
undefined	undefined	undefined
null	null	object ???

ŠTO JE JAVASCRIPT?

- Usporedba vrijednosti moguća je kroz provjere jednakosti
  - Sintaktički **implicitna (implicit, ==)** ili **eksplicitna (explicit, ===)**
- **Rezultat usporedbe uvijek daje boolean vrijednost**
- Uspoređivanje provodi **prisilno prilagođavanje tipova vrijednosti (type coercion)** kako bi se garantirala valjanost usporedbe
- **Tip varijable s desne strane se uvijek prilagođava tipu varijable s lijeve strane!**
- Rezultat prilagođavanja su **istinite (truthy) i neistinite (falsy) vrijednosti**
  - Falsy vrijednosti su `''`, `0`, `-0`, `NaN`, `null`, `undefined`, `false`

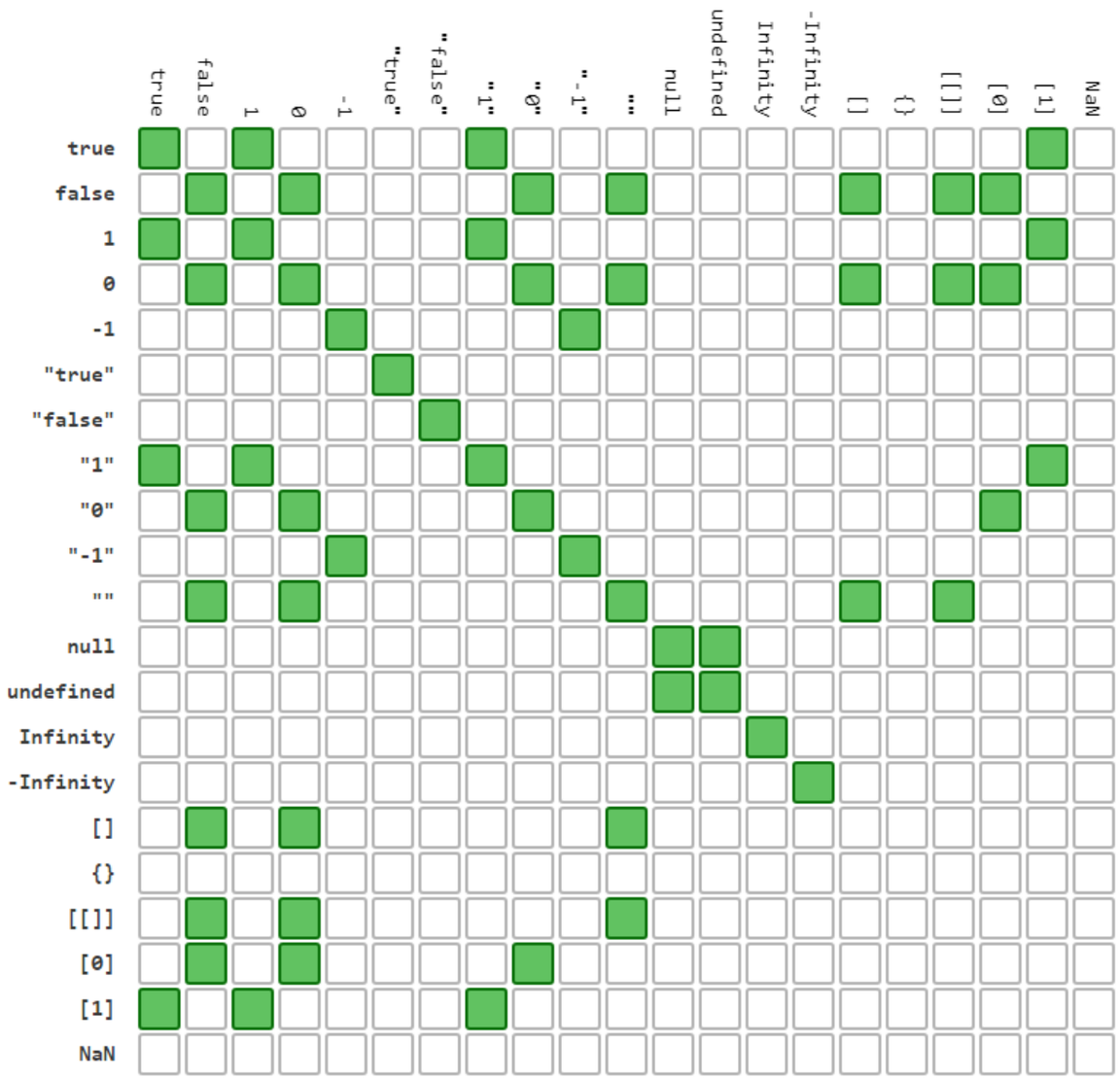
ŠTO JE JAVASCRIPT?

Smjernice za uspoređivanje nepoznatih vrijednosti:

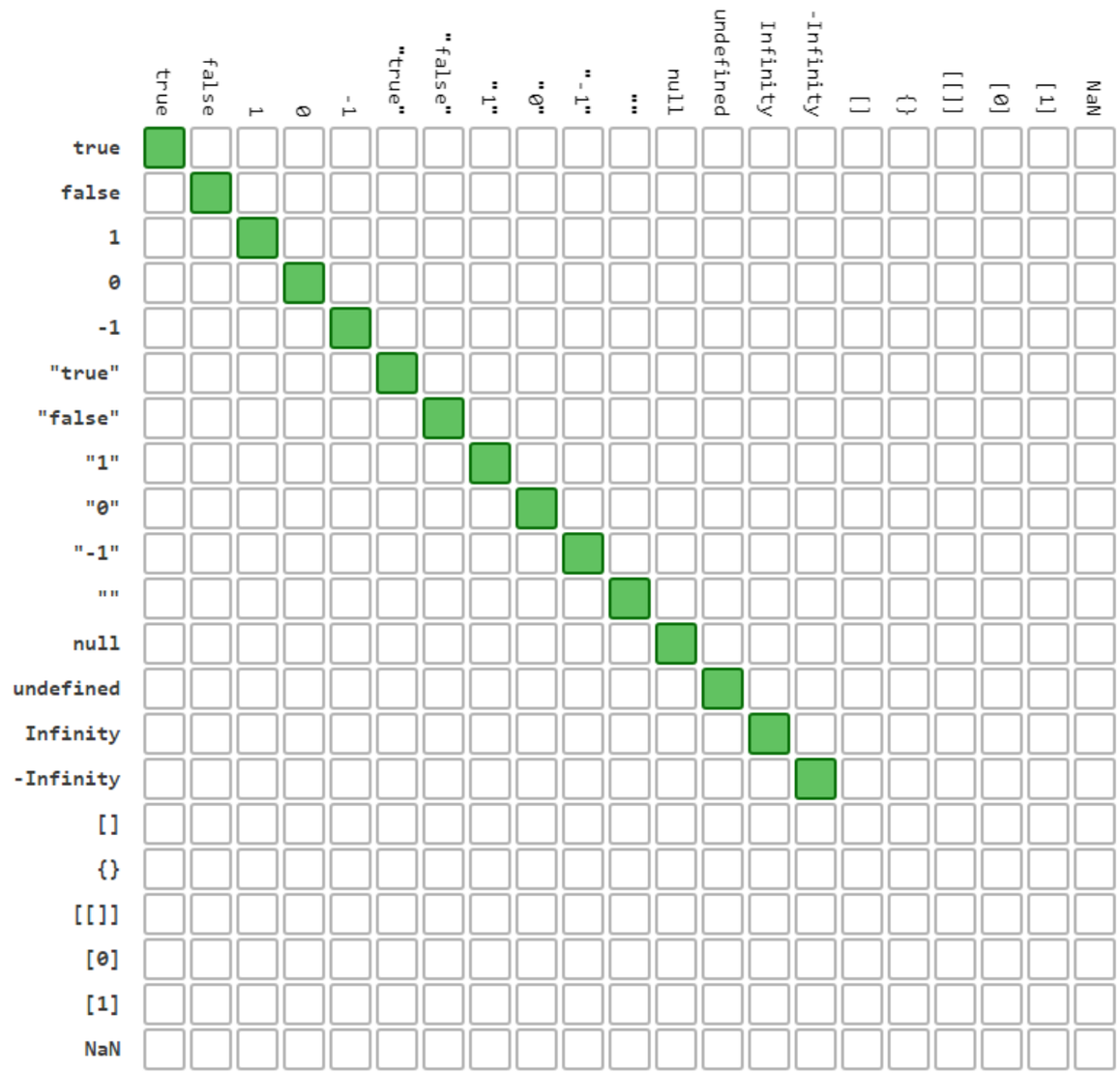
1) Ako bilo koja vrijednost može biti boolean tipa, koristite ===

2) Ako bilo koja vrijednost može biti 0, '', ili [], koristite ===

3) U svim ostalim slučajevima, koristite ==



Implicitno uspoređivanje (==)



Eksplicitno uspoređivanje (===)

https://dorey.github.io/JavaScript-Equality-Table/

Pitanje za milijun kuna

```
null > 0; // false

null == 0; // false


null >= 0; // ???
```

Pitanje za milijun kuna

```
null > 0; // false

null == 0; // false


null >= 0; // true
```

1. Call ToPrimitive(x, hint Number).

2. Call ToPrimitive(y, hint Number).

3. If Type(Result(1)) is String and Type(Result(2)) is String, go to step 16. (Note that this step differs from step 7 in the algorithm for the addition operator + in using 'and' instead of 'or'.)

4. Call ToNumber(Result(1)).

5. Call ToNumber(Result(2)).

6. If Result(4) is NaN, return undefined.

7. If Result(5) is NaN, return undefined.

8. If Result(4) and Result(5) are the same number value, return false.

9. If Result(4) is +0 and Result(5) is -0, return false.

10. If Result(4) is -0 and Result(5) is +0, return false.

11. If Result(4) is +∞, return false.

12. If Result(5) is +∞, return true.

13. If Result(5) is -∞, return false.

14. If Result(4) is -∞, return true.

15. If the mathematical value of Result(4) is less than the mathematical value of Result(5) --- note that these mathematical values are both finite and not both zero --- return true. Otherwise, return false.

16. If Result(2) is a prefix of Result(1), return false. (A string value p is a prefix of string value q if q can be the result of concatenating p and some other string r. Note that any string is a prefix of itself, because r may be the empty string.)

17. If Result(1) is a prefix of Result(2), return true.

18. Let k be the smallest nonnegative integer such that the character at position k within Result(1) is different from the character at position k within Result(2). (There must be such a k, for neither string is a prefix of the other.)

19. Let m be the integer that is the code point value for the character at position k within Result(1).

20. Let n be the integer that is the code point value for the character at position k within Result(2).

21. If m < n, return true. Otherwise, return false.

Abstract Relational Comparison Algorithm

(null > 0)

1. If Type(*x*) is different from Type(*y*), go to step 14.

2.If Type(*x*) is Undefined, return **true**.

3.If Type(*x*) is Null, return **true**.

4.If Type(*x*) is not Number, go to step 11.

5.If *x* is **NaN**, return **false**.

6.If *y* is **NaN**, return **false**.

7.If *x* is the same number value as *y*, return **true**.

8.If *x* is **+0** and *y* is **-0**, return **true**.

9. If *x* is **-0** and *y* is **+0**, return **true**.

10. Return **false**.

11.If Type(*x*) is String, then return **true** if *x* and *y* are exactly the same sequence of characters (same length and same characters in corresponding positions). Otherwise, return **false**.

12. If Type(*x*) is Boolean, return **true** if *x* and *y* are both **true** or both **false**. Otherwise, return **false**.

13.Return **true** if *x* and *y* refer to the same object or if they refer to objects joined to each other (see 13.1.2). Otherwise, return **false**.

14. If *x* is **null** and *y* is **undefined**, return **true**.

15. If *x* is **undefined** and *y* is **null**, return **true**.

16.If Type(*x*) is Number and Type(*y*) is String, return the result of the comparison *x* == ToNumber(*y*).

17.If Type(*x*) is String and Type(*y*) is Number, return the result of the comparison ToNumber(*x*)== *y*.

18. If Type(*x*) is Boolean, return the result of the comparison ToNumber(*x*)== *y*.

19. If Type(*y*) is Boolean, return the result of the comparison *x* == ToNumber(*y*).

20.If Type(*x*) is either String or Number and Type(*y*) is Object, return the result of the comparison *x* == ToPrimitive(*y*).

21.If Type(*x*) is Object and Type(*y*) is either String or Number, return the result of the comparison ToPrimitive(*x*)== *y*.

22. Return **false**.

Abstract Equality Comparison Algorithm

(null == 0)

if *null* < *0* is **false**, then *null* >= *0* is **true**

Greater-than-or-equal Operator

(null >= 0)

ŠTO JE JAVASCRIPT?

- **Varijable mogu pohraniti bilo koji tip vrijednosti** u bilo kojem trenutku izvođenja programa
- Varijable se deklariraju ključnim riječima `var`, `let`, i `const`, a ključna riječ diktira **opseg dostupnosti (scope)** varijable
- **Imena varijabli (identifier)** ne mogu biti ključne riječi i nevažeći znakovi

ŠTO JE JAVASCRIPT?

- **Funkcije odjeljuju dijelove programa u zasebne cjeline i opsege**
- Funkcije su **„građani prvog reda” (first-class citizen)** i mogu se proslijediti kao parametri drugim funkcijama te pohranjivati u varijable
- Funkcije se deklariraju ključnom riječi `function` ili skraćenom sintaksom `() => {}`
- Funkcija unutar sebe može deklarirati druge funkcije, pri čemu su argumenti roditeljske funkcije također dostupni potomku **(closure)**
- Deklaracija funkcija i varijabli s ključnom riječi `var` **povlače se na vrh trenutnog scopea** kako bi bile dostupne kroz ostatak programa **(hoisting)**
- Ako se unutar funkcije koristi ključna riječ `this`**, referencira se objekt koji ovisi o kontekstu pozivanja**, ne i sama funkcija

ŠTO JE JAVASCRIPT?

Smjernice za određivanje this reference:

1) U `strict mode`, referenca je `undefined`, u protivnom je globalni objekt (`window`)

2) Ako je funkcija dio objekta, referenca je na taj objekt

3) Ako je funkcija pozvana ili vezana uz objekt, referenca je na taj objekt

4) Ako se funkcija poziva s ključnom riječi `new`, referenca je na novi prazni objekt

```
function introduce() {
    console.log('Hello, I am ' + this.name);
}

const person = { name: 'Domagoj' };

introduce.call(person);
introduce.apply(person);
introduce.bind(person)();
// introduce(); -- error!

```

ŠTO JE JAVASCRIPT?

- **Objekti se mogu kreirati u deklarativnom (`{}`) ili konstruiranom (`Object()`) obliku**
  - Rezultat je isti, no razlika je u pristupu dodavanja svojstava na objekt
- Primitivi nisu objekti, ali se automatski **„zamataju" u odgovarajući objektni oblik (boxing)** kada se pokuša pristupiti njihovim svojstvima (npr. `'foobar'.toUpperCase()`)
- `null` i `undefined` nemaju svoj objektni oblik

ŠTO JE JAVASCRIPT?

- **Objekti posjeduju interno `[[Prototype]]` svojstvo, što je referenca ne neki drugi objekt**
  - Kor prototipnog nasljeđivanja, **objekti su povezani u prototipni lanac (prototype chain)**, zadnja karika u lancu je `Object.prototype` objekt
  - Prototipni lanac se prati kada sa pristupa svojstvu objekta
- **Nasljeđivanje u JavaScriptu se ne vrši kopiranjem ponašanja, nego referenciranjem**
- Operator `instanceof` provjerava prototipni lanac za prisutnost reference na traženi objekt (npr. `[] instanceof Array`)

ŠTO JE JAVASCRIPT?

- **JavaScript se smatra interpretiranim jezikom,** ali se u pravilu **prevodi za vrijeme izvođenja (Just-In-Time, JIT)** u sljedećim koracima:

  1) Tokeniziranje i leksiranje teksta s izvornim kôdom

  2) Obrada toka tokena u apstraktno sintaksno stablo

  3) Generiranje izvedbenog kôda

ŠTO JE JAVASCRIPT?

- Izvornim kôdom rukuju:

  1) Engine – odgovoran za prevođenje i izvođenje izvornog kôda

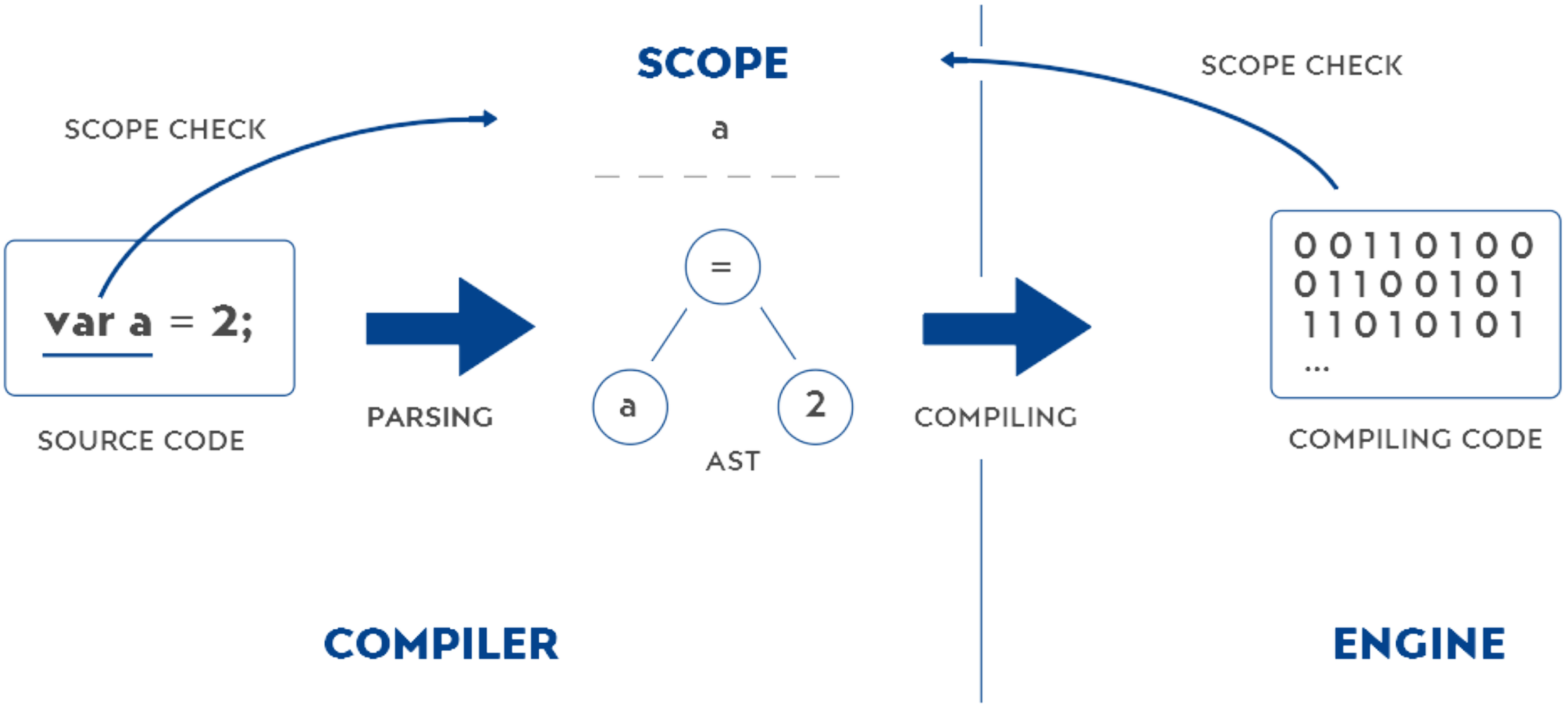
  2) Compiler – odgovoran za obradu izvornog kôda i generiranje izvedbenog kôda

  3) Scope – odgovoran za održavanje popisa deklariranih identifikatora i provođenje pravila pristupa resursima

- Scope radi provjeru s lijeve (left-hand side, **LHS**) i desne strane (right-hand side, **RHS**) strane operatora pridruživanja

  - LHS provjerava identifikatore, RHS provjerava vrijednosti i reference

ŠTO JE JAVASCRIPT?



„Developers, developers, developers, developers!”

Steve Ballmer, Ex-Microsoft CEO

JAVASCRIPT!?

- Revolucija JavaScripta potakla je konkurentne preglednike da razviju svoje implementacije jezika (npr. Microsoft JScript)
- **ECMA standardizacija** započinje u studenom 1996. od strane odbora s oznakom **TC39**
- **ECMAScript 1** utemeljen je na verziji JavaScripta iz lipnja 1997.
- **ECMAScript 2** (lipanj 1998.) definiran je kako bi se popravile razlike između ECMA i ISO standarda, jezik se ne mijenja
- **ECMAScript 3** (prosinac 1999.) uvodi regularne izraze, `do-while` petlju, iznimke i `try/catch` blokove, `in` i `instanceof` operatore, formatiranje brojeva pri ispisu, bolje rukovanje greškama

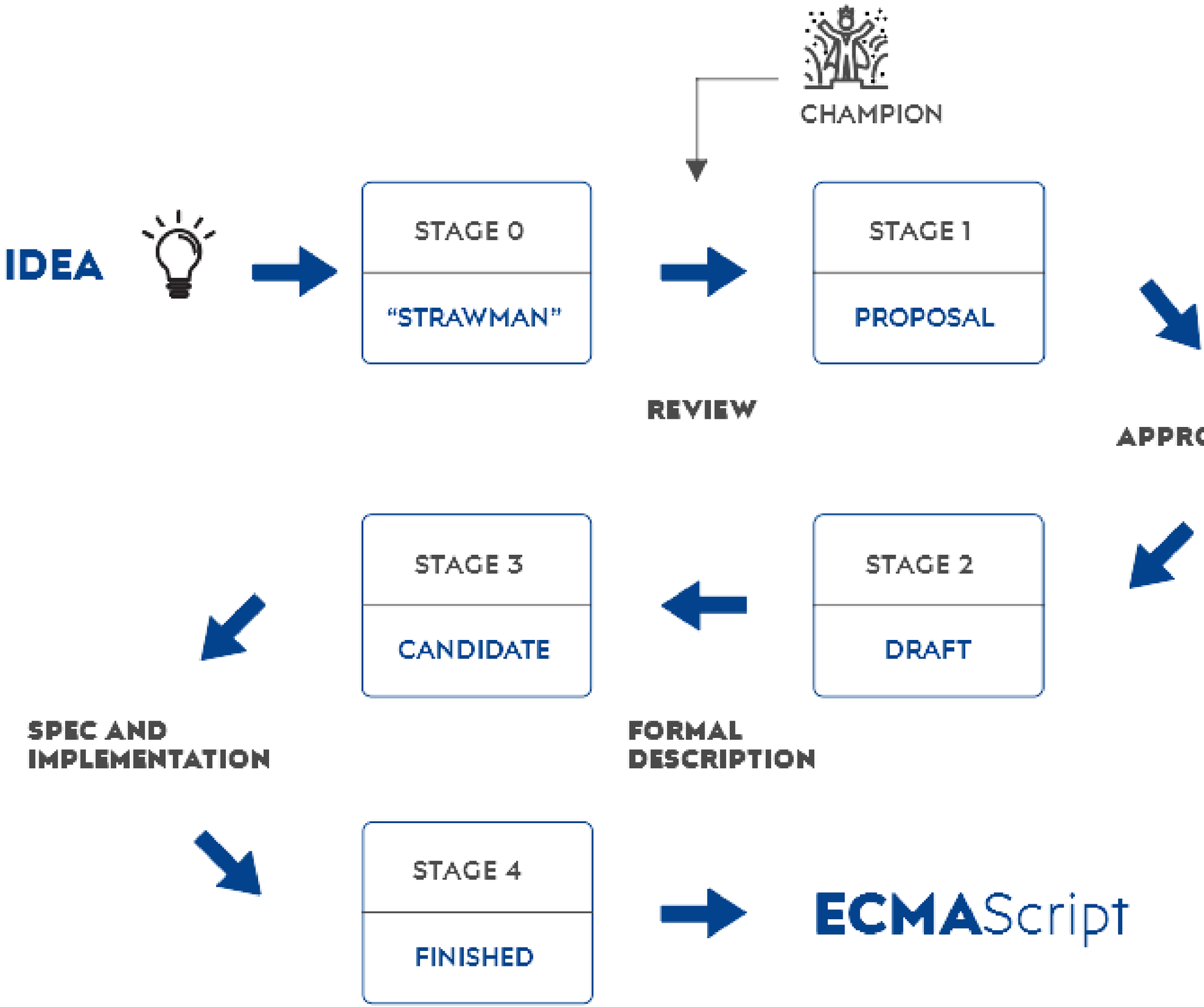
JAVASCRIPT!?

- **ECMAScript 4 nikada nije ugledao svjetlo dana**
- Inicijativa da se JavaScript pretvori u jezik opće namjene uvela je razdor među članove TC39 odbora zbog velikog opsega izmjena
- **Adobe**, **Mozilla**, **Opera**, i **Microsoft** već su odlučili većinu sintaksnih i funkcionalnih promjena kada je **Yahoo** na pregovore poslao svog predstavnika **Douglasa Crockforda**
- Crockfordovi snažni stavovi i argumenti protiv ECMAScript 4 uvjerili su Microsoftovog predstavnika da promjeni mišljenje
- **Microsoft** - čiji je Internet Explorer bio vodeći preglednik u to vrijeme - **odbio je prihvatiti bilo koju točku standarda**
- 2007. godine, nakon osam godina rada, **ECMAScript 4 se napušta**

JAVASCRIPT!?

- Nakon propasti ECMAScript 4, konsenzus TC39 odbora doveo je do izrade manjih, inkrementalnih verzija standarda
- ECMAScript 3.1 preimenovan je 2011. godine u **ECMAScript 5** i uvodi mnoge funkcionalnosti koje su temelj današnjim web aplikacijama
- **ECMAScript 6** (2015.) uspjeva uvesti nekoliko zahtjeva ECMAScript 4, zbog čega raste popularnost **transpiler** alata (**Babel**, **Traceur**)
- Iskustvo razvoja ECMAScript 4 potaklo je TC39 da uvede **proces predlaganja, evaluacije, i potvrđivanja novih funkcionalnosti** u kojem mogu sudjelovati svi članovi TC39 odbora
  - Proces je razbijen u „stadije zrelosti” kroz koje prijedlog funkcionalnosti prolazi
  - Kada prijedlog dođe do zadnjeg stadija, uključuje se u sljedeću verziju standarda

ECMAScript Proposal Process



JAVASCRIPT!?

- Novosti u standardu: **https://github.com/tc39/proposals**
- **Kako ostati u toku s promjenama?**
  - Dr. Axel Rauschmayer (**2ality.com**)
  - **ES.next News**
  - **JavaScript Weekly**
- **Gdje naučiti više?**
  - Axel Rauschmayer, „**Exploring JS**”
  - Marijn Haverbeke, „**Eloquent JavaScript**”
  - Kyle Simpson, „**You Don't Know JS**”
  - **Factory.hr GitHub repozitorij** s edukativnim materijalima

RADIONICA: JAVASCRIPT U PRAKSI

- **Mjesto:** Plava Tvornica, Lorenza Jägera 2, Osijek
- **Vrijeme:** Radionica #1 će se održati u četvrtak 14.6. u 17h
- **Ciljevi radionice:**
  - Postavljanje projekta pomoću NPM-a i Webpacka
  - Upoznavanje s trenutno najpopularnijim CSS preprocessorom (Sass)
  - Pisanje JavaScripta prema najnovijim standardima (ECMAScript 7)
  - Izrada manjeg reprezentativnog projekta u skladu s front-end trendovima
- **Prijave:** http://factory.hr/radionica

iOS DEVELOPER

WORDPRESS DEVELOPER

BACKEND DEVELOPER

FRONTEND DEVELOPER

Tražiš posao?

---

prijavi se na factory.hr/jobs

iOS DEVELOPER

WORDPRES DEVELOPER

BACKEND DEVELOPER

FRONTEND DEVELOPER

ANDROID DEVELOPER

MARKETING

PROJECT MANAGER

QA ANALYST

Tražiš praksu?

prijavi se na factory.hr/jobs

KONTAKT

Domagoj Štrekelj

domagoj.strekelj@gmail.com

https://github.com/dstrekelj

@dstrekelj

domagoj.strekelj

Hvala na pažnji!