

# DOMENSKO SPECIFIČNI MODELIRNI JEZIKI 2024/2025

8. predavanje

# Povzetek

- Vizualni jeziki
- Bločni jeziki
- Blockly
- Izdelava programskega okolja
- Demo

# Vizualni jeziki

## ■ Bločni jeziki (danes)

- Temeljijo na XML/JSON

## ■ Domensko specifični modelirni jeziki

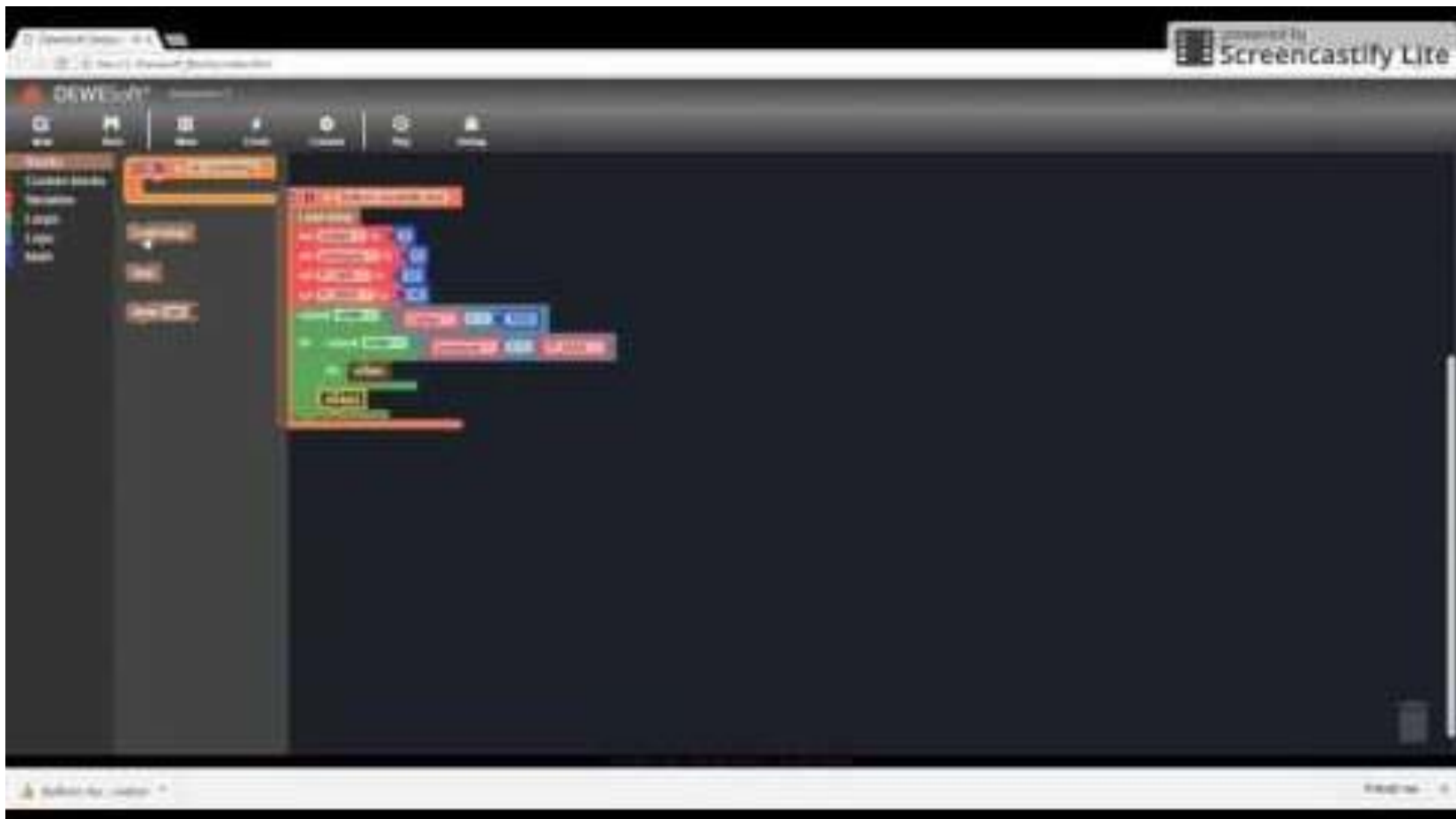
- Temeljijo na metamodelu

# Bločni jeziki

- Vizualni programski jeziki, kjer je koda sestavljena z uporabo **blokov namesto besedila**.
- Primeri: Blockly
  - Orodja: Scratch, App Inventor, tynker
- Uporaba
  - Povleci in spusti vmesnik.
  - Bloki se združujejo kot sestavljanke.
- Osnovni koncepti: logika, zanke, pogojni stavki, funkcije
- Spodbuja ustvarjalnost in eksperimentiranje.

# Primer bločnega jezika

## ■ Uporaba knjižnice Blockly



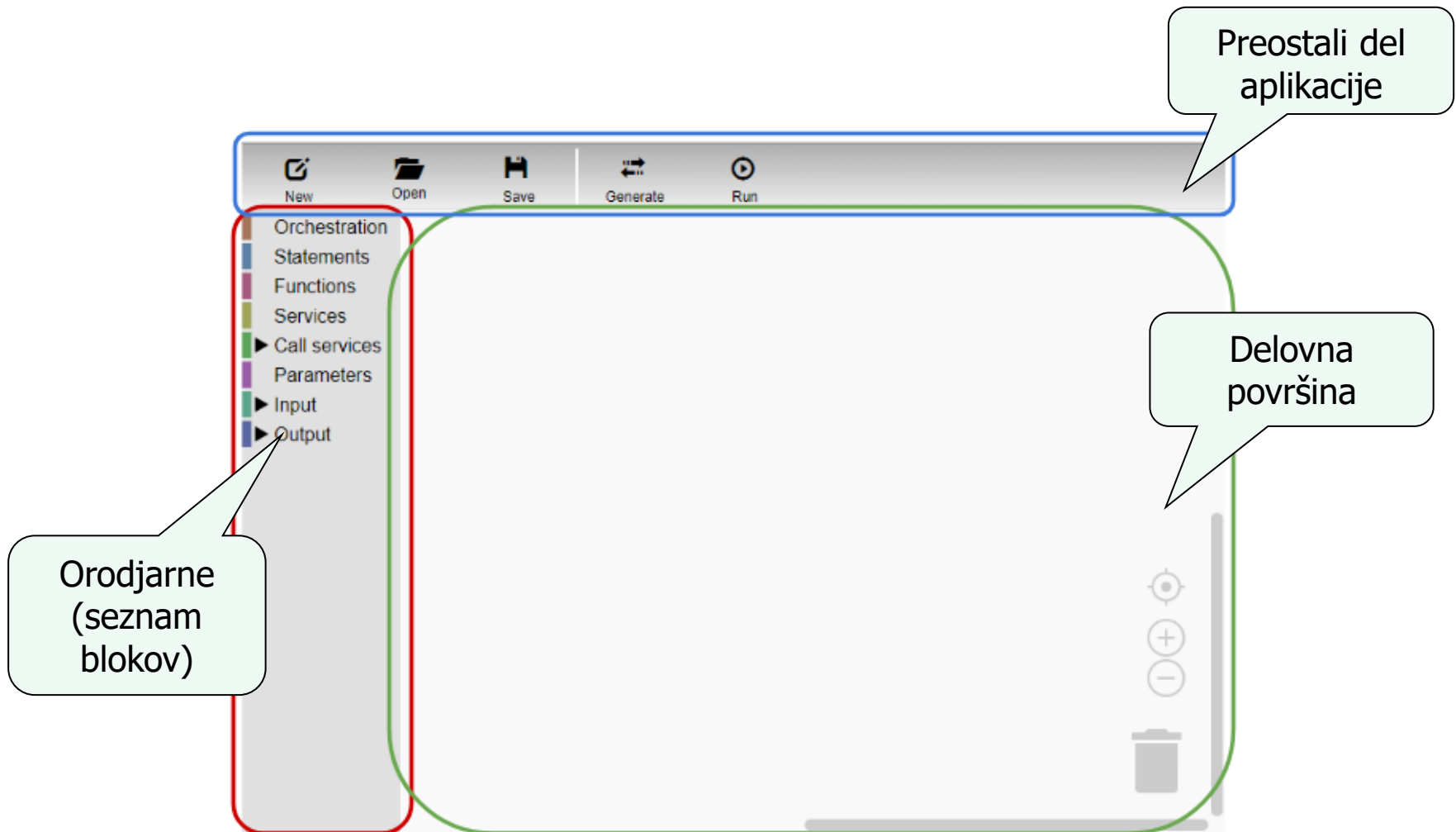
# Kaj je Blockly in kaj omogoča?

- Blockly je knjižnica za zapis vizualnih **blokovnih jezikov**.
- Omogoča vključitev **vizualnega urejevalnika kode** v spletno in mobilno (Android, iOS) aplikacije.
- Omogoča predstavitev **programskih konstruktov** v obliki prepletenih vizualnih blokov.
  - Uporaba obstoječih vizualnih blokov
    - Spremenljivke, logični izrazi, zanke, itd.
  - Definicija lastnih (domensko-specifičnih) vizualnih blokov
- Omogoča sestavljanje **sintaktično pravilnega programa**.

# Kako izdelati programsko okolje z Blockly-jem?

1. Vključitev urejevalnika Blockly
  - Urejevalnik Blockly sestoji iz **seznama blokov**, ki so uporabniku na voljo in iz **delovne površine**, na katero povlečemo posamezne bloke in jih sestavimo v program.
2. Ustvarjanje blokov
  - Za deloma **avtomatizirano** ustvarjanje blokov po meri, lahko uporabimo orodje [Blockly Developer Tools](#). Ustvarjene bloke nato v aplikaciji dodamo na seznam blokov. Lahko pa bloke ustvarimo **samostojno**.
3. Generiranje kode
  - Blockly sam po sebi je samo način, kako iz sestavljenih blokov **generirati kodo**.
4. Izdelava preostalega dela aplikacije
  - Ključni del aplikacije je, kako generirano kodo uporabiti.

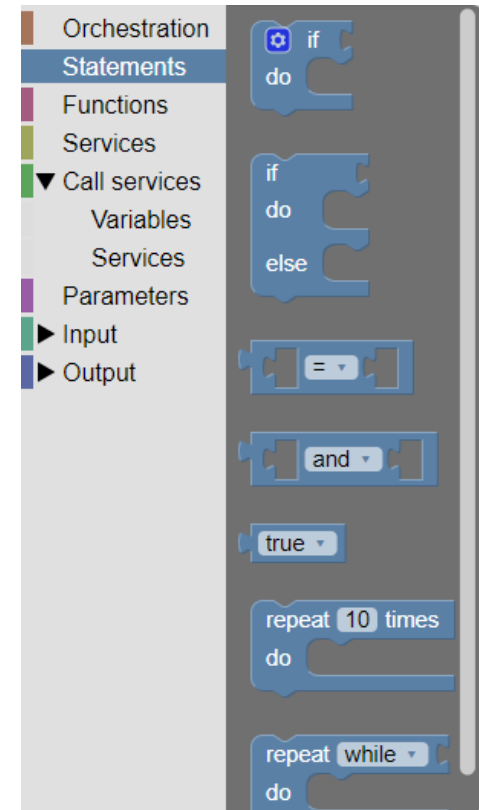
# Primer razvojnega okolja





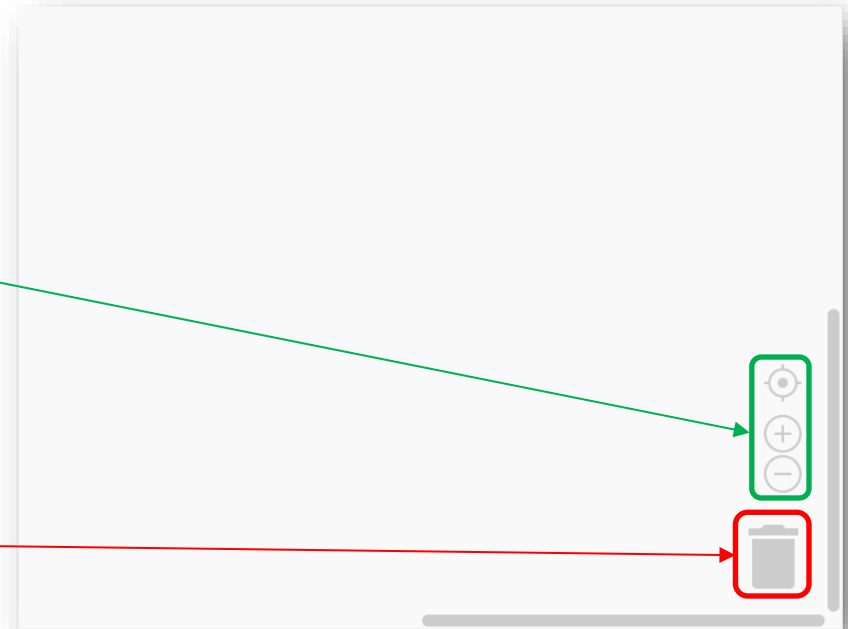
# Orodjarna (angl. toolbox)

- Vsebuje seznam blokov
- Nahaja se na levi strani aplikacije Blockly.
- Vsebuje nabor vseh blokov, ki jih lahko uporabimo pri gradnji programa.
- Vsaka **kategorija** (Orchestration, Statements, Functions, itd.) lahko vsebuje še **podkategorije** (Variables, Services).



# Delovna površina

- Namenjena je zlaganju blokov iz seznama v končni **program**.
- Blockly samodejno omogoči **funkcionalnosti** kot so povečevanje, pomanjšanje in centriranje sestavljenega programa na delovni površini.
- Namen koša je, da lahko posamezen blok ali skupek blokov odstranimo iz delovne površine.



# Orodje Blockly Developer Tools

- Omogoča **poenostavitve** pri konfiguraciji.
  - Omogočeno ustvarjanje, spreminjanje in shranjevanje blokov.
- **Generiranje notacije** posameznega bloka
  - JSON ali JavaScript
- Oblikovanje blokov z zlaganjem **različnih gradnikov**.
- Omogoča gradnjo seznama, kjer se nahaja nabor blokov in omogoča določitev **izgleda privzete delovne površine**.

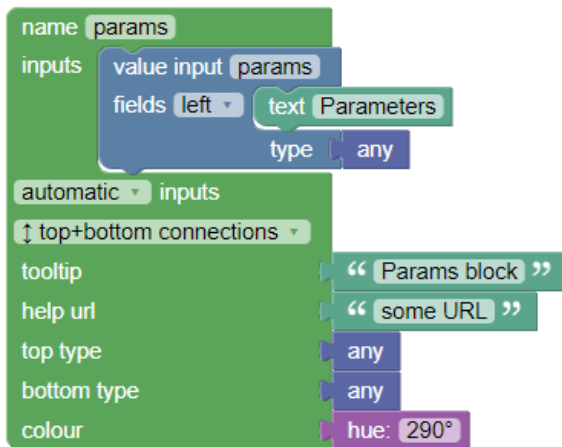
# Zapis vizualne notacije bloka

- Vsak blok mora imeti **unikatno ime**, s katerim se lahko nanj navezujemo.
- Izgled bloka opišemo z definicijo bloka, ki vključuje:
  - Besedilo
  - Barvo
  - Obliko
  - Opis, kako se drugi bloki lahko z njim povežejo

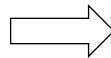


# Zapis vizualne notacije bloka - primer

## Blockly Developer Tools



Sestava bloka v orodju  
Blockly Developer Tools



Block Definition: JavaScript

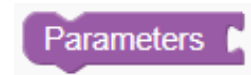
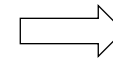
```
Blockly.Blocks['params'] = {  
  init: function() {  
    this.appendValueInput("params")  
      .setCheck(null)  
      .appendField("Parameters");  
    this.setPreviousStatement(true, null);  
    this.setNextStatement(true, null);  
    this.setColour(290);  
    this.setTooltip("Params block");  
    this.setHelpUrl("some URL");  
  }  
};
```

JavaScript definicija novega bloka

Block Definition: JSON

```
{  
  "type": "params",  
  "message0": "Parameters %1",  
  "args0": [  
    {  
      "type": "input_value",  
      "name": "params"  
    }  
  ],  
  "previousStatement": null,  
  "nextStatement": null,  
  "colour": 290,  
  "tooltip": "Params block",  
  "helpUrl": "some URL"  
}
```

JSON definicija novega bloka



Blok  
"params"

# Generiranje kode

- Bloki uporabljeni pri sestavljenem programu, se morajo pretvoriti v izvajalno kodo. Zato moramo **za vsak blok** zapisati **generator kode**.
- Izvajalna koda je lahko zapisana **v različnih jezikih** (JavaScript, Python, PHP, Lua, Dart, C#, itd.).
- Blok si mora pri generiranju kode pridobiti vse argumente in **podatke drugih blokov**, kateri so z njim združeni na vhodnem polju. V ta namen knjižnica Blockly vsebuje več funkcij:
  - *getFieldValue*.
  - *valueToCode*.
  - *statementToCode*.

# Generiranje kode - primer

- Vrstica 1: navežemo se na **ime bloka**, za katerega želimo zapisati generator kode. V našem primeru je to blok "params".
- Vrstice 2: s funkcijo *valueToCode* dostopamo do bloka "param", preko katerega **pridobimo generirano** kodo bloka, ki se nahaja na vhodu bloka "params". Kodo zapišemo v spremenljivko *value\_params*.
- Vrstica 5: **vrnemo** generirano kodo v spremenljivki *code*.

```
Blockly.JavaScript['params'] = function(block) {  
  var value_param = Blockly.JavaScript.valueToCode(block, 'param', Blockly.JavaScript.ORDER_ATOMIC);  
  // TODO: Assemble JavaScript into code variable.  
  var code = '...;\n';  
  return code;  
};
```

Generator kode za blok "params"

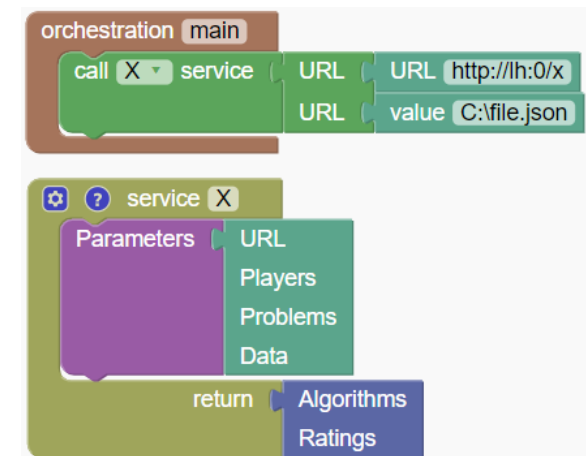


# Primer uporabe



# Vizualna notacija programa

- Program je sestavljen iz **dveh blokov**, ki lahko vsebujeta sklop drugih blokov in v generirani kodi predstavljata funkciji.
- Pri gradnji programa je pomembno, da pripnemo vsebino (bloke) vsem tistim blokom, ki so strukturirani tako, da imajo **definirane vhode**.
- V primeru, da bloka na vhodu ne pripnemo, bo pri generiranju kode, manjkalo del kode in **programa ne bo mogoče izvesti**.



Sestavljeni program

# Generirana koda

- Ko sestavimo željen program, ga lahko z uporabo aplikacije **pretvorimo v generirano kodo**.
- Pri generiranju, se koda, ki jo predstavlja posamezni blok v sestavljenem programu, **združi**.
- V tem primeru se koda generira v programski jezik C#.

```
static void Main(string[] args)
{
    var X = CallX("http://lh:0/x", "C:\\file.json");
}

static string CallX(string url, string json)
{
    WSX.MyXWebService ws = new WSX.MyXWebService();
    return ws.X(json);
}
```

Generirana programska koda C#



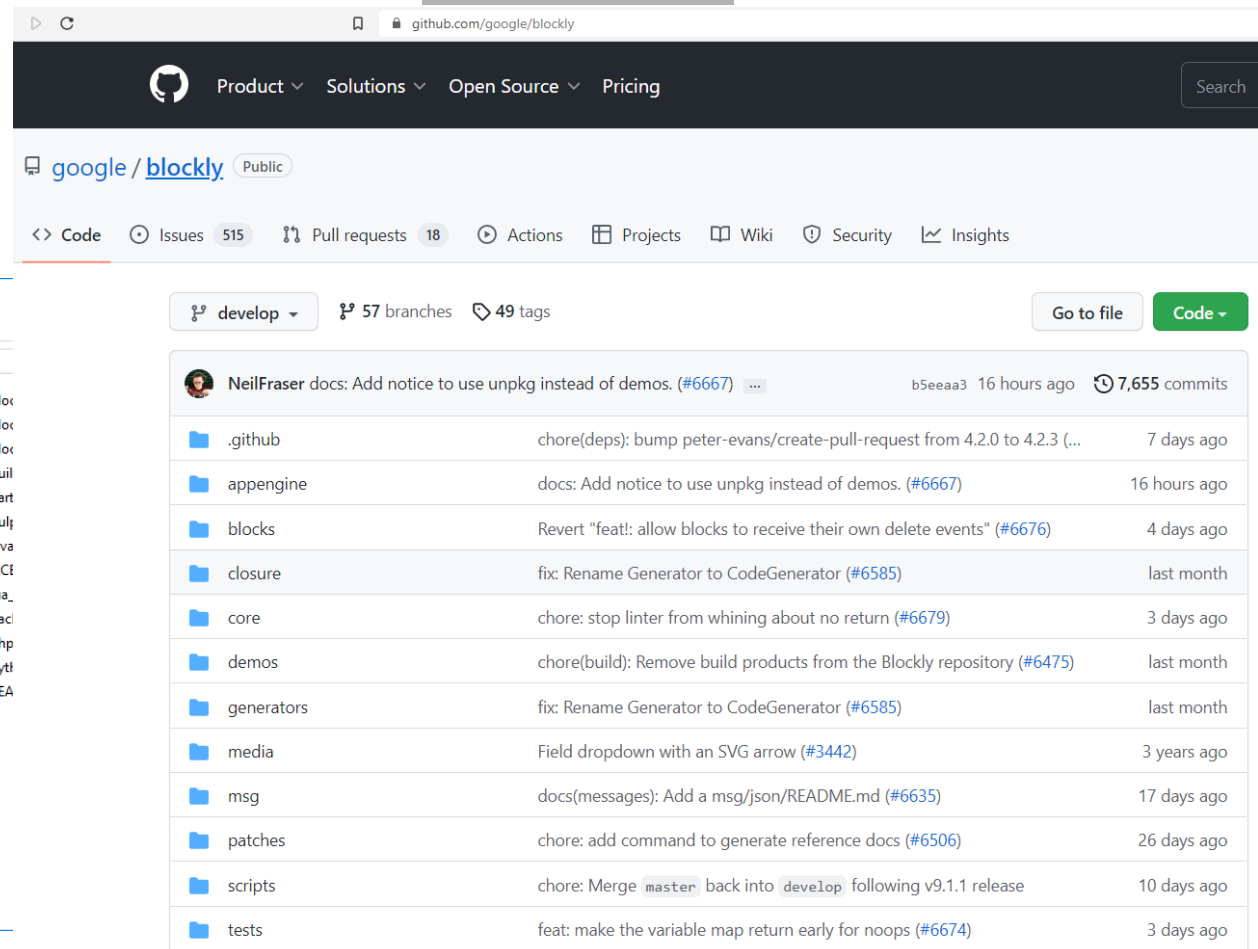
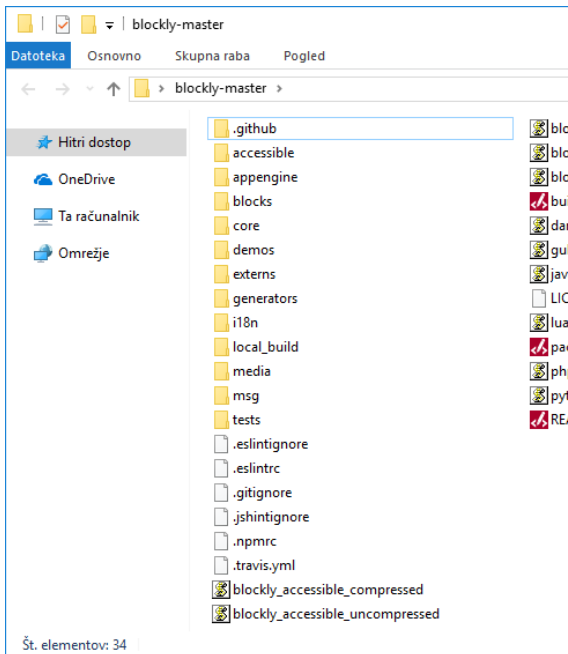
# Kako pričeti z razvojem bločnega jezika?

(Izdelava spletne aplikacije z uporabo knjižnice Blockly)



# Prenos izvorne kode Blockly

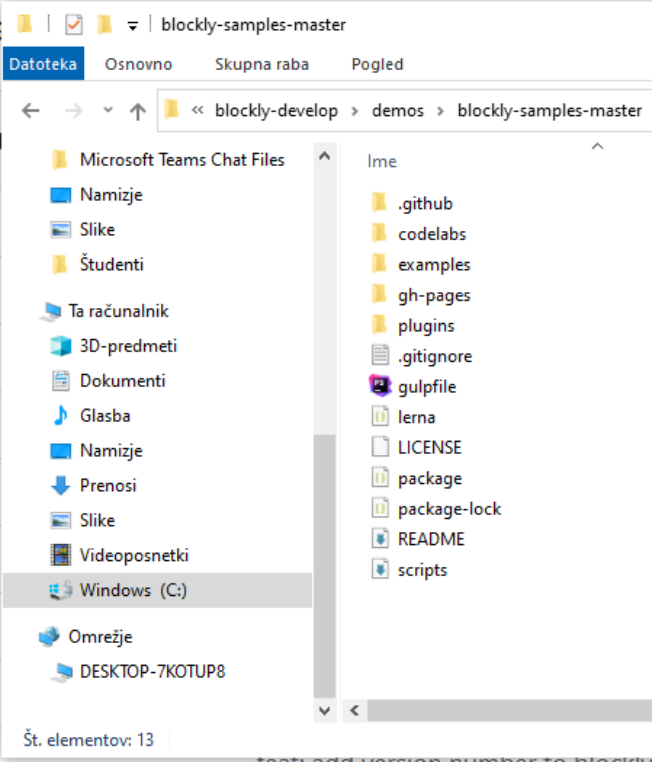
## ■ Prenos izvorne kode iz GitHub-a





# Prenos primerov Blockly

## ■ Prenos primerov iz [GitHub](#)



blockly-samples-master

Datoteka Osnovno Skupna raba Pogled

← → ↶ ↷ << blockly-develop > demos > blockly-samples-master

Microsoft Teams Chat Files

Namizje

Slike

Študenti

Ta računalnik

3D-predmeti

Dokumenti

Glasba

Namizje

Prenosi

Slike

Videoposnetki

Windows (C:)

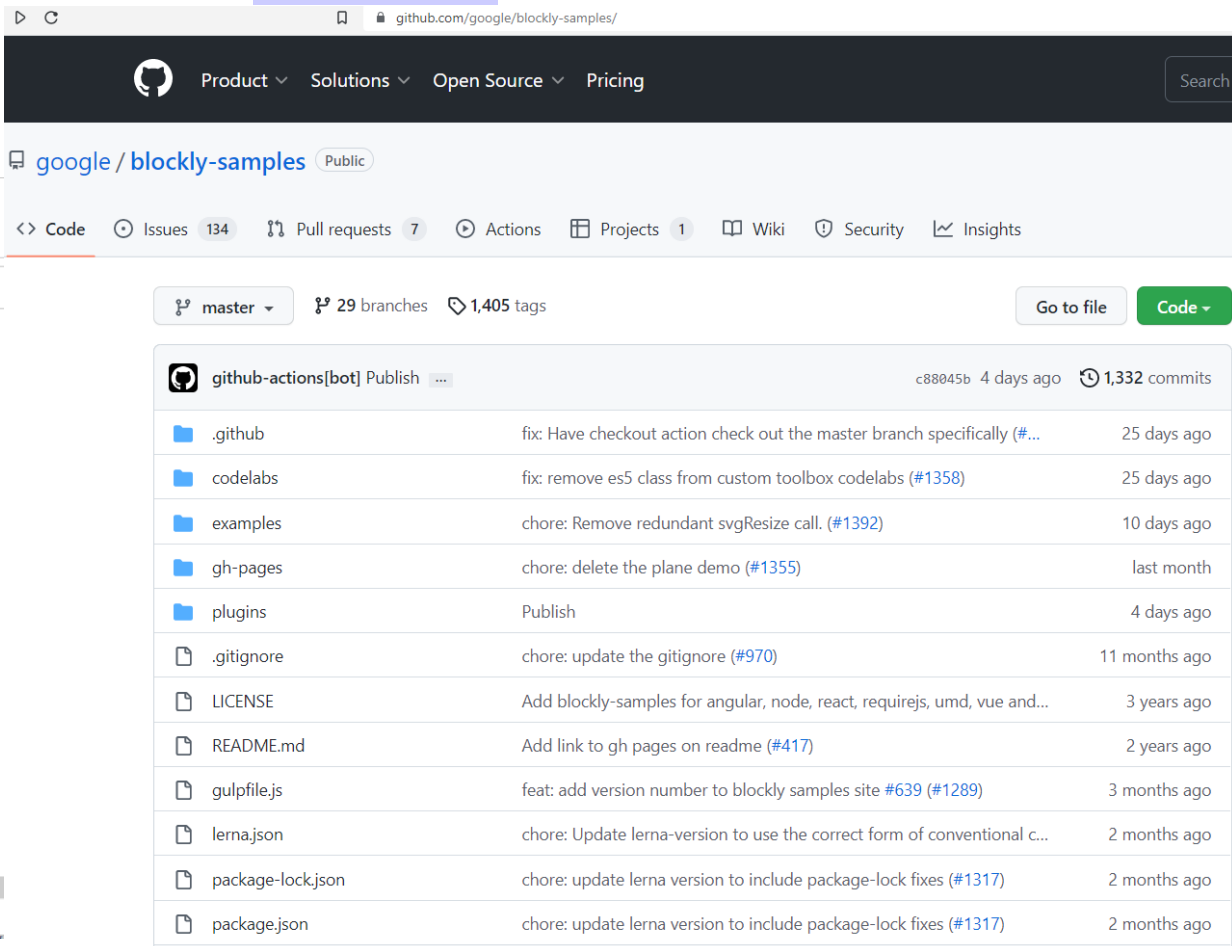
Omrežje

DESKTOP-7KOTUP8

Št. elementov: 13

Ime

- .github
- codelabs
- examples
- gh-pages
- plugins
- .gitignore
- gulpfile
- lerna
- LICENSE
- package-lock
- package-lock
- README
- scripts



github.com/google/blockly-samples/

Product Solutions Open Source Pricing

Search

google / blockly-samples Public

<> Code Issues 134 Pull requests 7 Actions Projects 1 Wiki Security Insights

master 29 branches 1,405 tags

Go to file Code

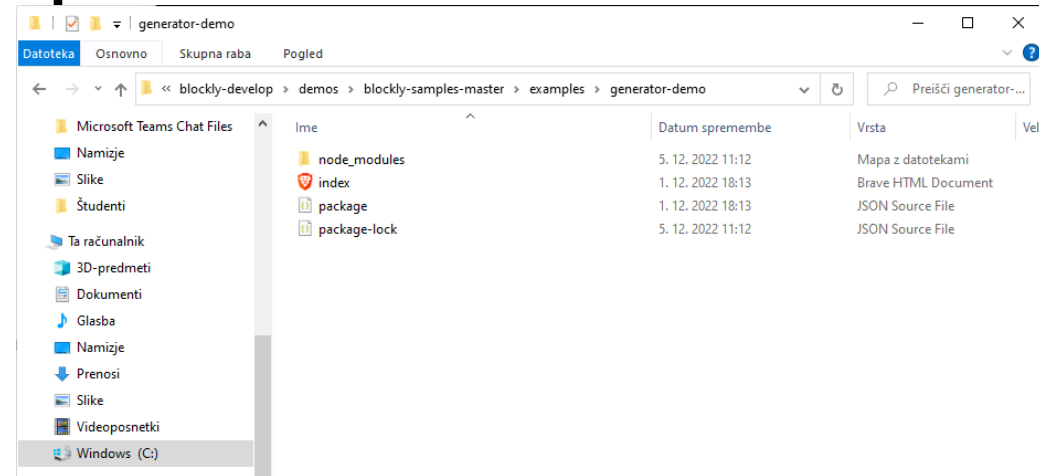
github-actions[bot] Publish c88045b 4 days ago 1,332 commits

.github	fix: Have checkout action check out the master branch specifically (#...	25 days ago
codelabs	fix: remove es5 class from custom toolbox codelabs (#1358)	25 days ago
examples	chore: Remove redundant svgResize call. (#1392)	10 days ago
gh-pages	chore: delete the plane demo (#1355)	last month
plugins	Publish	4 days ago
.gitignore	chore: update the gitignore (#970)	11 months ago
LICENSE	Add blockly-samples for angular, node, react, requirejs, umd, vue and...	3 years ago
README.md	Add link to gh pages on readme (#417)	2 years ago
gulpfile.js	feat: add version number to blockly samples site #639 (#1289)	3 months ago
lerna.json	chore: Update lerna-version to use the correct form of conventional c...	2 months ago
package-lock.json	chore: update lerna version to include package-lock fixes (#1317)	2 months ago
package.json	chore: update lerna version to include package-lock fixes (#1317)	2 months ago



# Zagon primerov uporabe

- Demo programi
- Npr. generator-demo



- Namestimo vse odvisnosti z *npm install*

```
C:\blockly-develop\demos\blockly-samples-master\examples\generator-demo>npm install
npm WARN deprecated request-promise-native@1.0.9: request-promise-native has been deprecated because it extends the now
deprecated request package, see https://github.com/request/request/issues/3142
npm WARN deprecated har-validator@5.1.5: this library is no longer supported
npm WARN deprecated uuid@3.4.0: Please upgrade to version 7 or higher. Older versions may use Math.random() in certain
circumstances, which is known to be problematic. See https://v8.dev/blog/math-random for details.
npm WARN deprecated request@2.88.2: request has been deprecated, see https://github.com/request/request/issues/3142

added 98 packages, and audited 99 packages in 3s

2 packages are looking for funding
  run `npm fund` for details

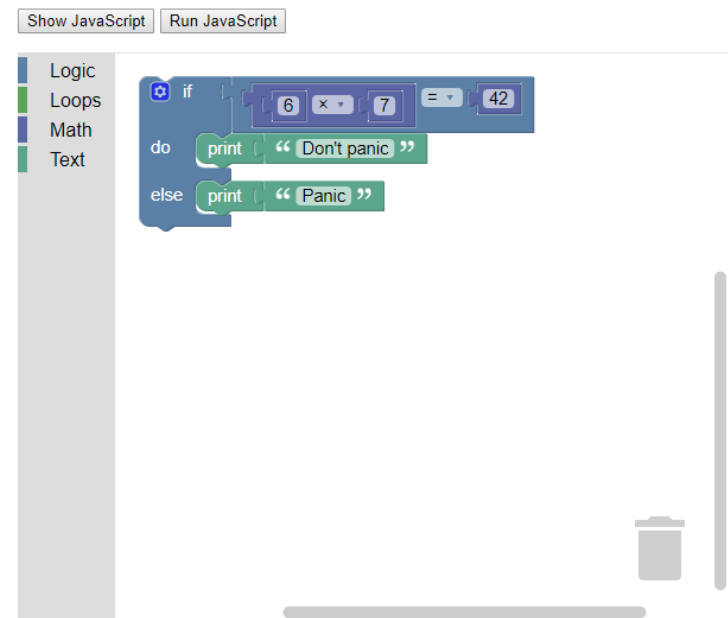
2 moderate severity vulnerabilities

To address all issues (including breaking changes), run:
  npm audit fix --force

Run `npm audit` for details.
```

# Zagon spletne aplikacije

- **Demo programi**
- Spletno aplikacijo odpremo v brskalniku  
*generator-demo/index.html*.
- Prikaže se nam spletna aplikacija z nekaj že **predpripravljenimi bloki** in s primerom sestavljenega programa.
- Iz sestavljenega programa lahko:
  - Generiramo JavaScript kodo
  - Generirano kodo zaženemo



Spletna aplikacija

# Struktura spletne aplikacije

- V datoteki *demos/generator/index.html* se nahajajo trije pomembnejši sklopi kode:

- **Vključevanje skript z** definicijami blokov in generatorji kode.
- Seznam **navedenih blokov**, ki jih uporabnik lahko uporabi pri sestavljanju programa.
- **Delovna površina** na kateri iz blokov sestavimo program.

```
<head>
  <meta charset="utf-8">
  <title>Blockly Demo: Generating JavaScript</title>
  <script src="../../blockly_compressed.js"></script>
  <script src="../../blocks_compressed.js"></script>
  <script src="../../javascript_compressed.js"></script>
  <script src="../../msg/js/en.js"></script>
  <style ...
</style>
</head>
```

```
<xml id="toolbox" style="display: none">
  <category name="Logic" colour="%{BKY_LOGIC_HUE}"> ...
  </category>
  <category name="Loops" colour="%{BKY_LOOPS_HUE}"> ...
  </category>
  <category name="Math" colour="%{BKY_MATH_HUE}"> ...
  </category>
  <category name="Text" colour="%{BKY_TEXTS_HUE}">
    <block type="text"></block>
    <block type="text_length"></block>
    <block type="text_print"></block>
  </category>
</xml>
```

```
<xml id="startBlocks" style="display: none"> ...
</xml>
```



# Spreminjanje vizualne notacije blokov

- **Bloke** spreminjamo/dodajamo/odstranimo v datotekah, ki se nahajajo v direktoriju *blocks/*.
- Paziti moramo, da ko delamo spremembe v datotekah, ki so v direktoriju *blocks/*, moramo ustrezno spremeniti tudi **skripte**, ki so vključene v datoteki *demos/generator/index.html*.
- `<script src="../../blocks_compressed.js"></script>`
- `<script src="../../generators/javascript/text.js"></script>`

# Spreminjanje generatorjev kode

- **Generatorje** kode za posamezen blok spreminjamo/dodajamo/odstranimo v datotekah, ki se nahajajo v direktoriju *generators/*.
- Tudi pri **generatorjih** kode moramo paziti, da imamo vključene ustrezne skripte.
- `<script src="../../javascript_compressed.js"></script>`
- `<script src="../../generators_javascript/text.js"></script>`



# Demo

## ■ Vključevanje Blockly knjižnic

```
<> index.html X JS movements.js
<> index.html > html > head > style > body
1  <!DOCTYPE html>
2  <html>
3  <head>
4    <meta charset="utf-8">
5    <title>DSMJ example</title>
6    <script src="./node_modules/blockly/blockly_compressed.js"></script>
7    <script src="./node_modules/blockly/blocks_compressed.js"></script>
8    <script src="./node_modules/blockly/javascript_compressed.js"></script>
9    <script src="./node_modules/blockly/msg/en.js"></script>
10   <script src="movements.js"></script>
11   <script src="movements-gen.js"></script>
```

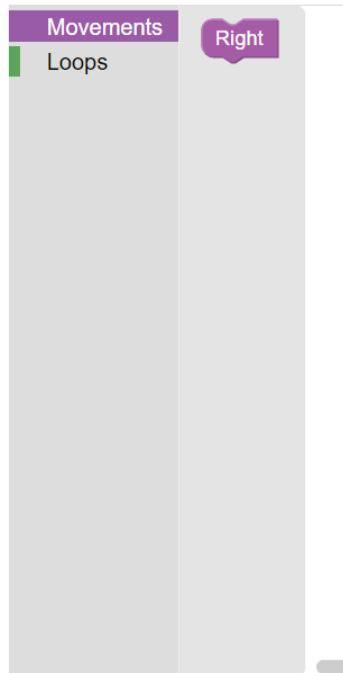


# Demo

## ■ Definiranje lastnih blokcev

DSMJ example: Moving

Show JavaScript Run JavaScript



```
<> index.html JS movements.js X
JS movements.js > ...
1  'use strict';
2
3  Blockly.defineBlocksWithJsonArray([
4  {
5      "type": "moveRight",
6      "lastDummyAlign0": "CENTRE",
7      "message0": "Right",
8      "previousStatement": null,
9      "nextStatement": null,
10     "colour": 300,
11     "tooltip": "Move right",
12     "helpUrl": ""
13   }
14   ]);
```



# Demo

## ■ Generiranje kode

DSMJ example: Moving objects with right command

Show JavaScript Run JavaScript

Movements  
Loops

Right

repeat 10 times  
do Right

```
var x=0;
var y=0;
for (var count = 0; count < 10; count++) {
  x = x + 1;
}
```

```
function runCode() {
  // Generate JavaScript code and run it.
  window.LoopTrap = 1000;
  Blockly.JavaScript.INFINITE_LOOP_TRAP =
    'if (--window.LoopTrap == 0) throw "Infinite loop.";\\n';
  var code = 'var x=0;\\n';
  code += 'var y=0;\\n';
  code += Blockly.JavaScript.workspaceToCode(demoWorkspace);
  Blockly.JavaScript.INFINITE_LOOP_TRAP = null;
  code += 'window.alert(\\'x: \\ ' + x + '\\ y: \\ ' + y);';
  try {
    eval(code);
  } catch (e) {
    alert(e);
  }
}
```

index.html JS movements-gen.js X

```
JS movements-gen.js > ...
1 'use strict';
2
3 Blockly.JavaScript['moveRight'] = function(block) {
4   return 'x = x + 1;\\n';
5 };
6
```

# Vaje

## ■ Naloga 8.1. (obvezna)

- Za lasten jezik ustvarite vizualno notacijo v Blockly-ju. Naloga je vredna 4 točke.

## ■ Naloga 8.2 (neobvezna)

- Za nalogo 8.1. zapišite generacijo kode v JavaScript-u in jo tudi ovrednotite.
  - Generiranje kode vsebuje npr. novo pozicijo po izvedenem programu ter kontrolo ali je bila naloga opravljena. (1 točki)
  - Vizualizirajte/animirajte sprehod figure od začetne do končne pozicije. (3 točke)

# Tekmovanje Pišek

- Naloga 8.1. po vzoru iz tekmovanja
  - <https://pisek.acm.si/contents/4907-4902/>
- Se želi kdo priključiti pripravi nalog za tekmovanje?

# Vprašanja

