

Hacking Decidim

```
-/projects/Decidim/decidim (0.19-stable)$ bundle exec rake development_app
  create
  create  README.md
  create  Rakefile
  create  .ruby-version
  create  config.ru
  create  .gitignore
  create  Gemfile
    run  git init from "."
S'ha inicialitzat un dipòsit buit del Git en /home/ivan/projects/Decidim/decidim/development_app/.git/
  create  package.json
  create  app
  create  app/assets/config/manifest.js
  create  app/assets/javascripts/application.js
  create  app/assets/javascripts/cable.js
  create  app/assets/stylesheets/application.css
  create  app/channels/application_cable/channel.rb
  create  app/channels/application_cable/connection.rb
  create  app/controllers/application_controller.rb
  create  app/helpers/application_helper.rb
  create  app/jobs/application_job.rb
  create  app/mailers/application_mailer.rb
  create  app/models/application_record.rb
  create  app/views/layouts/application.html.erb
  create  app/views/Layouts/mailер.html.erb
  create  app/views/layouts/mailер.text.erb
```

Follo

PRO-TIP! You're not a hacker if you don't have your terminal in black

About me: Ivan Vergés

- Telecommunication Engineering (ETSETB/UPC)
- Working as a freelance developer and system maintainer since 2004
- PHP/Python/Ruby/Javascript
- 100% Linux & Free Software
- Lead developer in Platoniq since 2018
- I play the guitar



<https://twitter.com/ivanverges>



<https://meta.decidim.org/profiles/microstudi>



<https://github.com/microstudi>



ivan@platoniq.net

Decidim experience

- The unofficial guide to install Decidim:

<https://platoniq.github.io/decidim-install/>

- Adapting Decidim for organizations
- Custom modules (direct verifications)

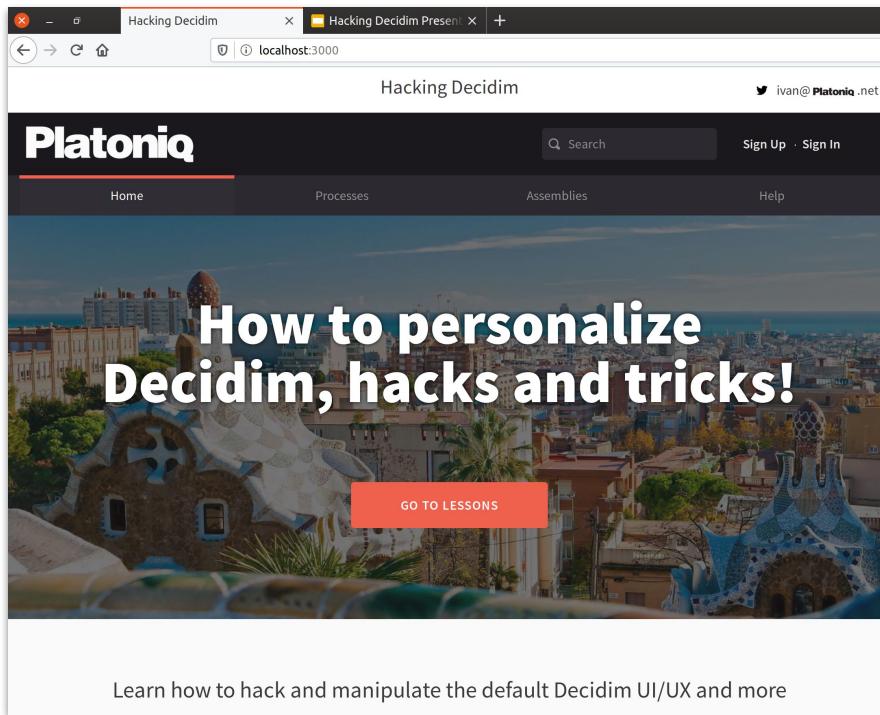
<https://github.com/Platoniq/>

- Among the contributors to the Decidim Source Code
- In the Team of maintainers of the source code

<https://github.com/decidim/decidim/graphs/contributors>

Requirements: <https://github.com/platoniq/decidim-hacks>

Docker up & Running with the test site ready
`docker-compose up`



A proper text editor

<https://code.visualstudio.com/>

The screenshot shows the Visual Studio Code interface with the following details:

- Title Bar:** blog-post.js - gatsby-graphql-app - Visual Studio Code - Insiders
- File Explorer:** EXTENSIONS: MARKETPLACE. Shows installed extensions: Python (2019-6-24221), Microsoft (Linting, Debugging (multi-threaded,...)), GitLens - Git supercharged (9.8.2), C/C++ (0.24.1), ESLint (1.9.0), Debugger for Chrome (4.11.7), Microsoft Language Support (0.47.0), and VS Code Icons (9.2.0).
- Editor:** JS blog-post.js (active tab). The code is a Gatsby component that imports `graphQL` from `gatsby`, `React`, and `Image`. It defines a default export function that takes `data` as an argument and returns a JSX object. The JSX includes a `div` with `dataFormat` and `debug` props, a `button` with `debugger` and `decodeURI` props, an `Image` component with `decodeURIComponent` prop, and a conditional rendering block for `defaultStatus`.
- Terminal:** Shows the command `yarn dev` being run, with output indicating compilation was successful in 79ms at 11:40:46 AM.
- Bottom Status Bar:** master, 0:0.11, 0:0.6, 1:0, 1:0, LF, JavaScript, and a status bar icon for Pine Wu.

Install reminder

<https://code.visualstudio.com/>

<https://github.com/Platoniq/decidim-hacks/blob/master/docs/install-docker-on-windows-10-home.md>

<https://github.com/Platoniq/decidim-hacks/blob/master/docs/install-docker-on-macos.md>

```
git clone https://github.com/Platoniq/decidim-hacks.git
```

```
git pull
```

```
docker-compose up
```

Decidim-hacks is self-contained: <http://localhost:3000/processes>

localhost:3000/processes

HIGHLIGHTED PROCESSES

Basic Hacking examples (level 1)

This lesson covers some basics customizations.

More info

Hacking technique examples (level 2)

This lesson covers 5 basic techniques to change the default look & feel and behaviours of Decidim.

More info

title (WORKSPACE)

- decidim
- decidim-hacks
- app
- > assets
- > channels
- > controllers
- > helpers
- > jobs
- > mailers
- > models
- > views
- > layouts
- decidim
 - > _head_extra.html.erb
 - > _head_platonic.html.erb

```
Creates a sticky header-->
<!-- Topbar -->
<div class="title-bar">
  <? if current_organization.official img header? ?>
    <img alt="Logo" />
  <? else ?>
    <div><? if current_organization.name ?>
      <? current_organization.name ?>
    </div>
  </? else ?>
</div>
<div class="row column topbar">
  <div class="logo-wrapper">
    <? render partial: "layouts/decidim/logo" , locals: { organization: current_organization } ?>
    <div style="margin-top:5px">Cool tweaks</div>
  </div>
</div>
<? render partial: "layouts/decidim/topbar_search" ?>
<? render partial: "layouts/decidim/language_choose" ?>
<div class="hide-for-medium topbar_menu">
  <button type="button" data-toggle="offCanvas">
```

TAKE PART

TAKE PART

Find all the exercises here

Prepare environment

- A browser Tab with <http://localhost:3000/>
- A browser Tab with <https://github.com/Platoniq/decidim-hacks>
- Visual Studio Code opened, open the folder where you downloaded Decidim-hacks

Goals: what are we going to learn

- Get to know Decidim internally:
 - Directory structure, modular structure
- How to personalize Decidim in the frontend:
 - Change html, css and javascript defaults by our own implementations
- How to interact with the database directly:
 - Access the rails console
- How to change some backend behaviours:
 - Monkeypatch classes (overwrite methods)

Wilms, Ledner and Swift

Inici Procesos Assemblees Iniciatives Consultes Jornades Ajuda

Eaque minima quidem enim quae.

PARTICIPA

Culpa eius nobis nihil dolores ut est vero magnam recusandae blanditiis harum aspernatur sit exercitationem.

REGISTRAT ➤

PROCESOS ACTIUS

Optio ex non expedita laudantium.
Est nihil eum saepe architecto.
Fas actua Ut impedit.

VEURE TOTS ELS PROCESOS

ASSEMBLEES ACTIVES

Ut aut id fugit dolore.
Eius aut amet labore deleniti.
Ut vel voluptatem molestiae voluptas.
Querat totam officia ratione cupiditate.

Wilms, Ledner and Swift

Inici Procesos Assemblees Iniciatives Consultes Jornades Ajuda

Optio ex non expedita laudantium.
Est nihil eum saepe architecto.
Fas actua Ut impedit.

FASE 1 DE 3
Est et.
23-09-2019 - 23-12-2019
VEURE LES FASES

MES ➔

Minima praesentium quam.
Repudiandae maxime illum. Nemo culpa rerum. Persipciatis voluptatum saepe.

DOCUMENTS RELACIONATS

LIBERO ALIQUAM.. PDF 20 KB
Corporis Itaque voluptates tempora impedit.

IN [1 DOCUMENT]

IMATGES RELACIONADES

ILLUM VELET.

ILLUM VELET.
13 Novembre 2019 - 10:55:14:55

QUI RATIONE.

QUI RATIONE.
13 Novembre 2019 - 10:55:14:55

Referència: I-PART-2019-10-2
Compartir ➔
Incrusta ➔

How to turn this?

Voting is open 1-31.10.2019 - You can vote here ►

Helsinki Home Areas Information Search English Sign in

OmaStadi

Participatory budgeting makes your ideas and voice heard

OmaStadi – Helsinki's participatory budgeting



Through the Ruuti Budget, young people can annually come up with ideas and negotiate things that concern them. The Ruuti Budget offers an opportunity to affect the development of youth services, leisure activities and the entire City.

Youth Ruuti Budget



OmaStadi is Helsinki's way of doing participatory budgeting. Helsinki uses €4 million annually for realising ideas developed by the residents of the city. You can bring up ideas and vote on [omastadi.hel.fi](#).

[Vote!](#)

Vote and make a difference!

Between 1-31.10.2019, residents of Helsinki get to vote on OmaStadi plans made by the residents. The plans with the most votes will be realised.

Voting is open 1-31.10.2019 - You can vote here ►

Helsinki Home Areas Information Search English Sign in

OmaStadi

Western Helsinki

#OmaStadi · Reijola, Munkkiniemi, Haaga, Pitäjänmäki and Kaarela

PHASE 6 OF 7 Voting 2019-10-01 - 2019-10-31 View phases Röster

INFO **BUDGETS** **PLANS** **PROPOSALS 2018** **EVENTS**

The OmaStadi major district of Western Helsinki includes the neighbourhoods of Reijola, Munkkiniemi, Haaga, Pitäjänmäki and Kaarela. Between 15 November and 9 December 2018, you can submit proposals on how the sum of €615,200 allocated for this area should be spent. What would make the district more functional, pleasant and fun?

Every year, the City of Helsinki will allocate €4 million of its budget to be decided upon by the city's residents. The city is divided into seven major districts, and each district has been allocated a budget according to its population size. What would make the district more functional, pleasant and fun? You are not required to have accurate cost information in the proposal phase. The ideas are later developed into feasible plans by the residents and experts from the City services. The experts will create cost estimates for the plans. These plans are then voted on. The purpose of OmaStadi is to draw up proposals and make plans that are equal for all and benefit everyone. The district's proposals can be voted on by everyone aged 12 or over. Submitting proposals, developing plans and voting are all done in the OmaStadi.hel.fi service.

The major district of Western Helsinki includes the following neighbourhoods:

- Reijola: Lasko, Vanha Ruskeasuo, Pikkupuisto, Mellalohi
- Munkkiniemi: Niemenmäki, Munkkivuori, Talinranta, Vanha Munkkiniemi, Kuutisaari, Lehtisaari
- Haaga: Etelä-Haaga, Kivihaka, Poljois-Haaga, Lassila, Pikkupuisto
- Pitäjänmäki: Tali, Pajamäki, Pitäjämäki business zone, Reimarila, Marttila, Koritala
- Kaarela: Kannelmäki, Maununneva, Malminkartano, Hakuninmäki, Kuninkaantammi, Honkasuo.

Related images



Into this...

Plaza Pública

Español | Register | Entrar

El espacio donde nos encontramos para decidir.

Plaza Pública es la plataforma de participación de la Ciudad de México donde la ciudadanía puede proponer, debatir, votar y decidir proyectos y disposiciones que afectan su vida en la ciudad.

Consultas
Vota en los temas puestos a consulta por parte del gobierno
[Ir a consultas](#)

Asambleas
Próximamente podrás involucrarte en un grupo de decisión temática.
[Ir a asambleas](#)

Iniciativas
Pronto podrás crear propuestas y firmas para mejorar la ciudad
[Ir a iniciativas](#)

¿Qué hacemos con la feria?

Las personas que habitanos y disfrutamos la Ciudad de México podemos opinar sobre qué tipo de parque de diversiones queremos tener en el Bosque de Chapultepec.

Las ideas y propuestas sobre qué tipo de atracciones debería tener, qué debería conservarse, qué hacer con la icónica montaña rusa o para qué sector de la población debería estar pensado, servirán de guía para las bases del concurso de licitación internacional que se abrirá próximamente para decidir el futuro de ese preciado espacio de la Ciudad.

El espacio de Chapultepec donde está la feria,
¿debe seguir siendo un parque de diversiones?



Si
172 votos

No
33 votos

Última actualización: 2019-10-21 10:35 hrs

Procesos terminados



Programa de Gobierno de la Ciudad de México 2019-2024



Diagnóstico Ciudadano
¿Cómo es la LIMPIEZA EN EL CENTRO HISTÓRICO?



Convocatoria para que formen parte del Consejo Consultivo de la Comisión de Reconstrucción



Elección de la persona titular de la Comisión de Búsqueda

Español | Register | Entrar

Plaza Pública

Programa de Gobierno de la Ciudad de México 2019-2024

#ProgramaDeGobiernoCDMX • Participa en la construcción de una Ciudad de innovación y derechos.

FASE 3 DE 3
Incorporación de comentarios
2/6/2019 - 1/7/2019
[VER LAS FASES](#)

EL PROCESO PROGRAMA DE GOBIERNO 2019-2024

Conoce y comenta a favor, neutral o en contra de las acciones del Programa de Gobierno 2019-2024.

El Gobierno de la Ciudad de México pone a discusión pública el Programa de Gobierno 2019-2024 con el propósito de compartir y recoger la visión de la ciudadanía en la elaboración del documento que establece en los ejes, objetivos y acciones a realizar.

Este programa fue construido con las voces, plumas, opiniones de muchas personas y por supuesto, escuchando las opiniones de cientos de miles de ciudadanas y ciudadanos. Hoy nos toca gobernar para todos y para todas y por ello se pone a disposición de los habitantes de la Ciudad para su consulta y discusión.

La consulta estará abierta del 12 al 30 de abril.

Conocer la opinión y propuestas de la ciudadanía es clave en la búsqueda de consensos en torno a la construcción de la ciudad de innovación y derechos que queremos lograr.

I DOCUMENTOS RELACIONADOS

RESULTADOS DE LA CONSULTA PDF 1000 KB
Resultados de la Consulta del Plan de Gobierno de la Ciudad de México 2019-2024

PROGRAMA DE GOBIERNO DE LA CIUDAD DE MÉXICO 2019-2024 PDF 20 MB
Programa de Gobierno de la Ciudad de México 2019-2024

PROGRAMA DE GOBIERNO 2019-2024 VER TODO (133)

VER TODO (133)

2.1.6. Mejorar la protección social para el desempleo
Propuesta oficial

Implementar mejoras en las políticas públicas de protección social dirigidas a las personas en...

10/04/2019 ▲ SEGUIR ↗ 0 ■ 0

5.1.8. Estrategia de seguridad para el transporte público
Propuesta oficial

Objetivo Mejorar las estrategias para disminuir la incidencia delictiva en el transporte público...

10/04/2019 ▲ SEGUIR ↗ 0 ■ 1

1

▲ Seguir

QUÉ SE DECIDE
Las acciones que integran el Programa de Gobierno.

QUIÉN PARTICIPA
Habitantes de la Ciudad de México

CÓMO SE DECIDE
Comentando a favor, neutral o en contra de la propuesta.

ÁMBITO
Estatal

GRUPO PROMOTOR
Jefatura de Gobierno

FECHA DE INICIO
12 de abril de 2019

FECHA DE FINALIZACIÓN
1 de julio de 2019

Referencia: Plaza Pública-PART-2019-04-01
[Compartir](#) ↗
[Incrustar](#) ↘

TUUSULA

Kirjaudu sisään Rekisteröidy Hae

Etusivu Prosessit Tietoa

Tervetuloa kehittämään Tuusulaa

OSALLISTUMISALUSTA

Aktiiviset prosessit

OSALLISTU

Osallistuva budjetointi 2020 Aktiivinen valihe Ideahaku

Osallistuva budjetointi Tuusulassa 2019 Aktiivinen valihe Tulosten julkaisu

NÄYTÄ KAikki PROSESSIT

Kuinka osallistun?

TAPAHTUMAT
Selvitä missä ja milloin voit osallistua julkisille tapahtumiin.

KESKUSTELUT
Vilaltele, keskustele ja jaa näkemyksiä rikastuttaaksesi sinua koskettavia aiheita.

IDEAT
Tee ideoita, tue muiden tekemää ideoita ja edistä muutoksia, joita haluat nähdä.

LISÄTIEDOT

Tervetuloa Tuusulan osallistumisalustalle!

Vantaa OSALLISUUSALUSTA

Hae Rekisteröidy Kirjaudu sisään

Etusivu Osallistu Tietoa

TERVETULOA KEHITTÄMÄÄN YHDESSÄ KORSON SUURALUETTA!

OSALLISTU

MEIDÄN KORSO -HANKKEESSA KEHITETÄÄN KORSON SUURALUEEN TURVALLISUUTTA JA YHTEISÖLLISYTTÄ.
TULE MUKAAN IDEOIMAAN, KESKUSTELEMAAN, ÄÄNESTÄMÄÄN JA TOTEUTTAMAAN!

REKISTERÖDY >

TULOSTEN JULKISTUSJUHLA

Mitä Korson suuralueella tehdään keväällä 2019 aikana turvallisuuden ja yhteisöllisyyden parantamiseksi? Korsojaiset ovat pääseet vaikuttamaan hellimäksi alkupuolella äänestämällä, mikä idea Meidän Korso -hankkeessa saat toteutusraaha.

Juhlimme voittajaidoista yhdessä keskiviikkona 6.3. klo 17 aikaan Luron Villanikassa. Juhlassa kuulet voittajaidoista ja muiden toteuttamisesta sekä halutessasi voit ilmoittautua mukaan jonkin idean toteuttamiseen.

OSALLISTU

Goals: what are we NOT going to learn

- To code/program:
 - Ruby, Css, Javascript or Html
- Ruby & Rails
- To install Decidim

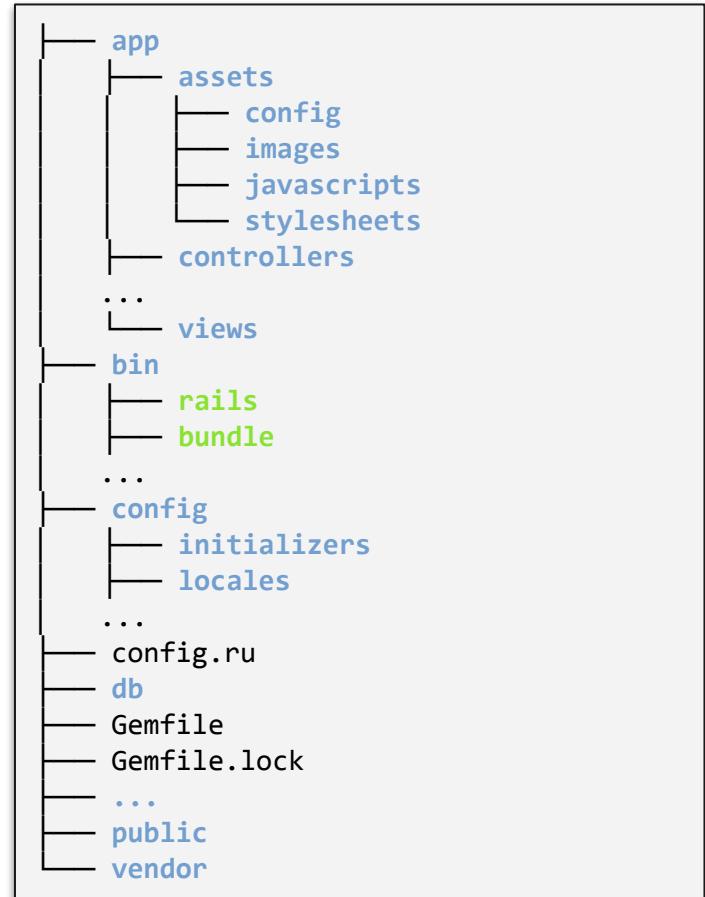
What is Decidim

- It's a **Gem** for the **Ruby and Rails** framework
- It is a **Generator** (like `rails new`).
- **It generates our app.**

```
~ $ decidim my_application
      create
      create README.md
      create Rakefile
      create .ruby-version
      create config.ru
      create .gitignore
      create Gemfile
      run git init from ".."
```

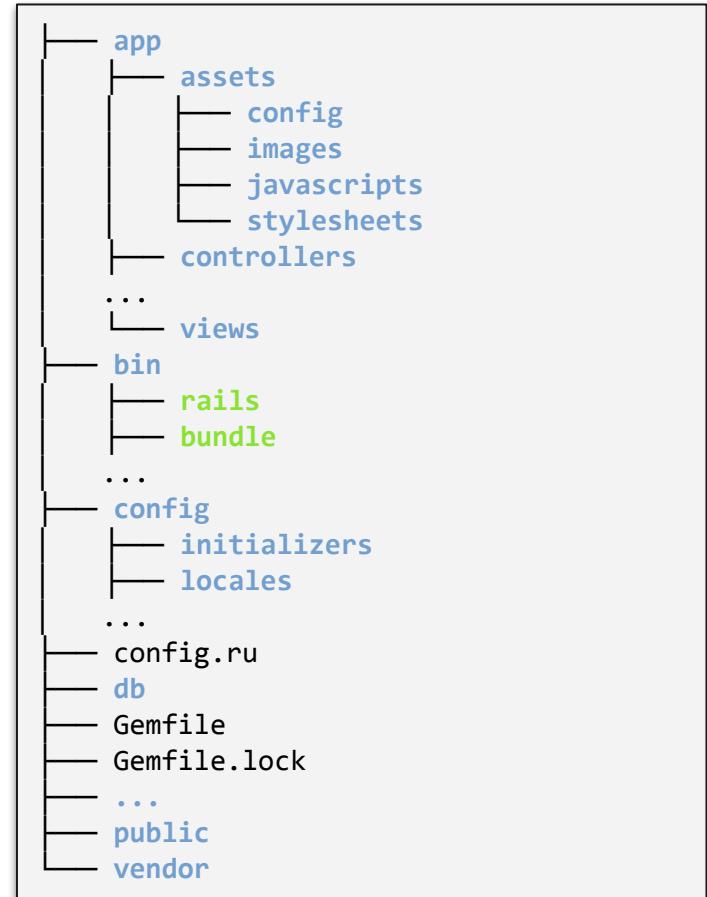
Decidim is Ruby & Rails

- **Gemfile** is the file used by **Bundler** to download all specified packages (gems) and its dependencies.
- **Gemfile.lock** contains all the gems installed with the exact version.
- **bundle**
 - installs Gems specified in the Gemfile
 - runs commands in the context of the Gemfile
- **rails**
 - Runs commands to manage Ruby & Rails:
 - bin/rails server
 - bin/rails console
 - bin/rails db:migrate

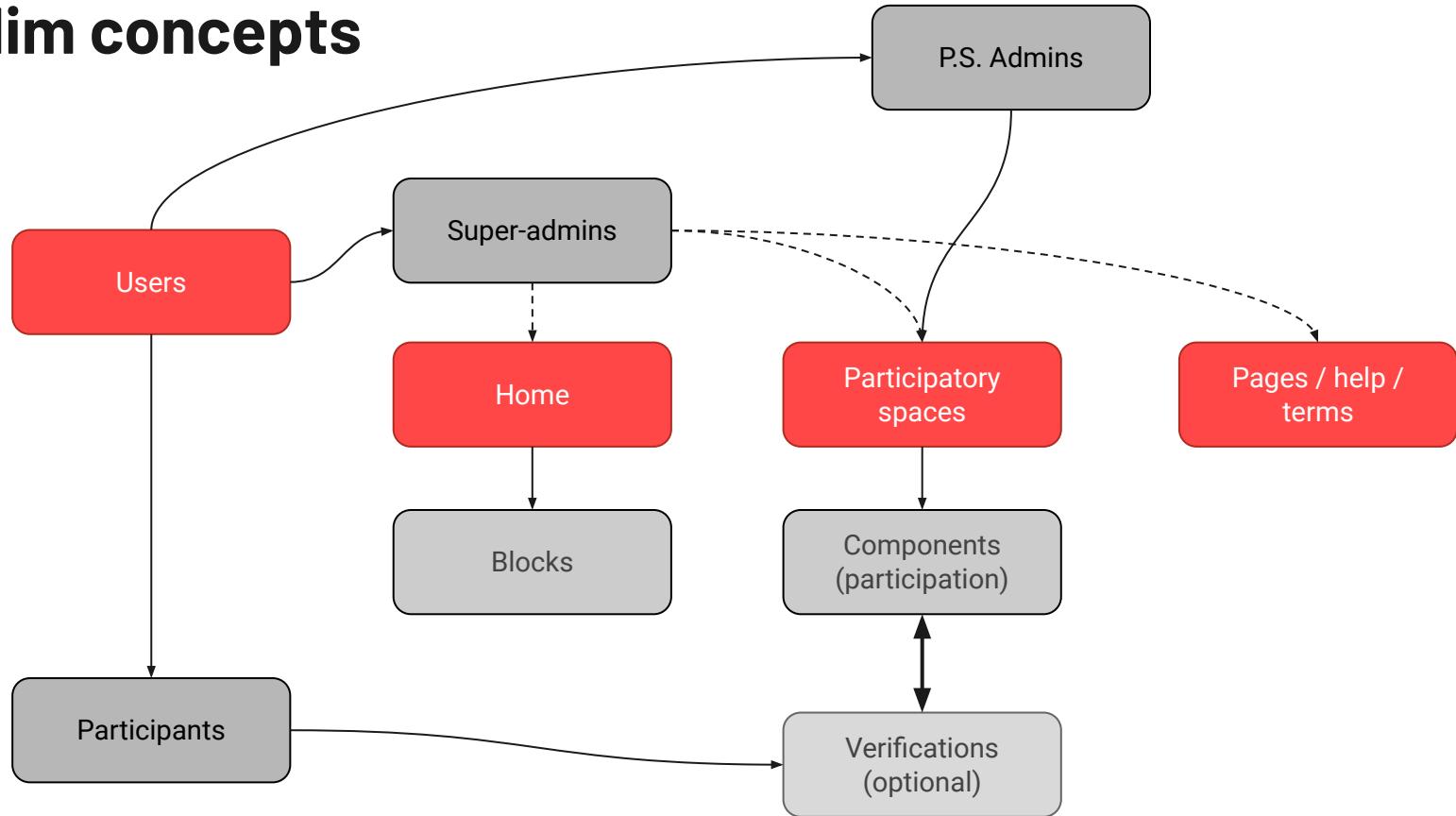


Decidim is Ruby & Rails

- **Rails** is (very) opinionated: it assumes there's one way to do most of the common things in the web.
- **"It's magic"**: it wants you to write the minimum amount of code. It's ok to not understand what's going on.
- It's for the Web and it's REST.
- Folder structure is immediately recognizable:
- In this workshop:
 - `app/assets/images`
 - `app/assets/javascripts`
 - `app/assets/stylesheets`
 - `views`
 - `config/initializers`
 - `config/locales`



Decidim concepts



Operational design for participation

P. Features

- Info
- **Phases**
- Categories
- Admins
- Components

- Info
- Categories
- **Members**
- Admins
- Components

Participatory spaces

Processes

Assemblies

Initiatives

...

Components

Proposals

Meetings

Surveys

Debates

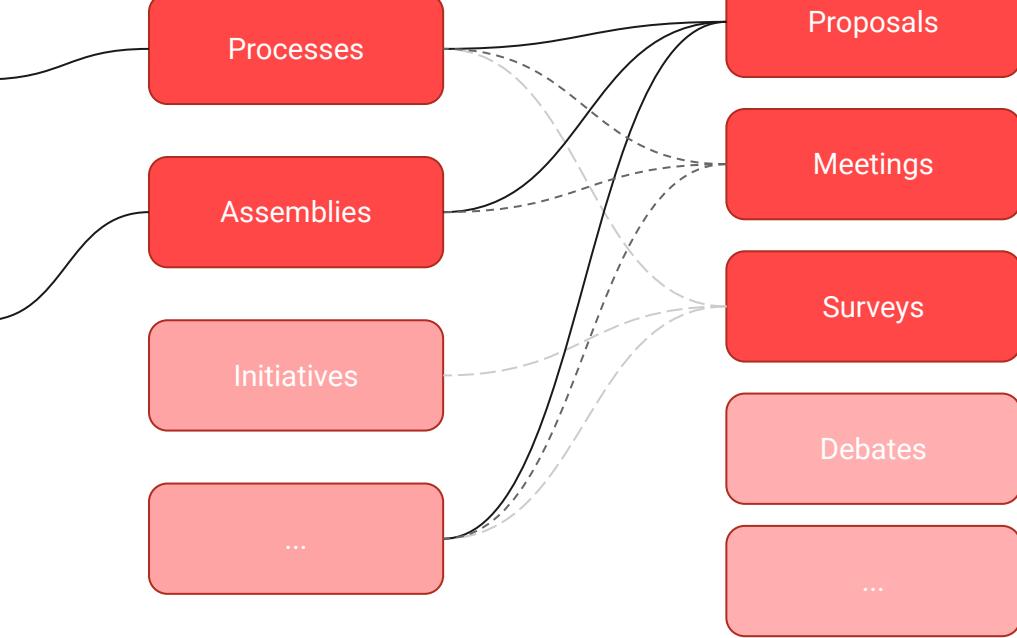
...

C. Features

- Comments
- **Endorsements**
- Votes
- ...

- Comments
- **Geolocation**
- Minutes
- ...

- Questions
- ...

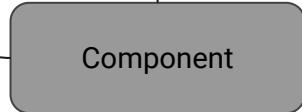
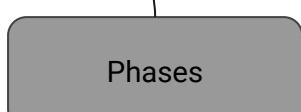


Co-design Platoniq's and ICLEI&FRIENDS's Sessions at Berlin's Consortium Assembly !

#openheritage · Co-design some of the activities in the next meeting of the consortium

 PHASE 1 OF 2

Proposal definition
2019-04-14 - 2019-05-09

[VIEW PHASES](#)[THE PROCESS](#)[DIGITAL COMMUNITY & PARTICIPATION](#)[IMPROVING SOCIAL MEDIA ACTIVITIES](#)[SOCIAL SURVEY FOR LABS](#) Info Component Phases

<https://labs.openheritage.eu/processes/berlin-meeting>

CO-DESIGN PLATONIQ'S AND ICLEI&FRIENDS'S SESSIONS AT BERLIN'S CONSORTIUM ASSEMBLY !

COMPONENTS

ADD COMPONENT ▾

| COMPONENT NAME | COMPONENT TYPE | ACTIONS |
|-----------------------------------|----------------|---------|
| Digital community & participation | Proposals | |
| Improving social media activities | Proposals | |
| Social Survey for Labs | Survey | |

COMPONENTS

View public page

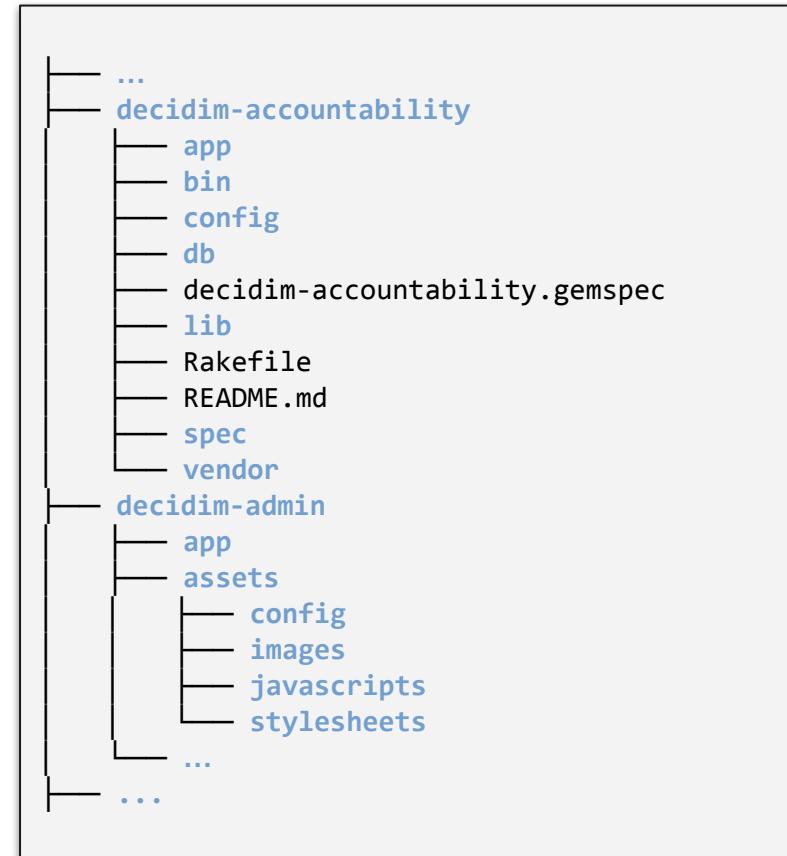
- Dashboard
- PROCESSES**
- Info
- Phases
- Components**
- Digital community & participation 5
- Improving social media activities 9
- Social Survey for Labs
- Categories
- Attachments
- Folders
- Files
- Process admins
- Private participants
- Moderations

Process admin

Decidim modules

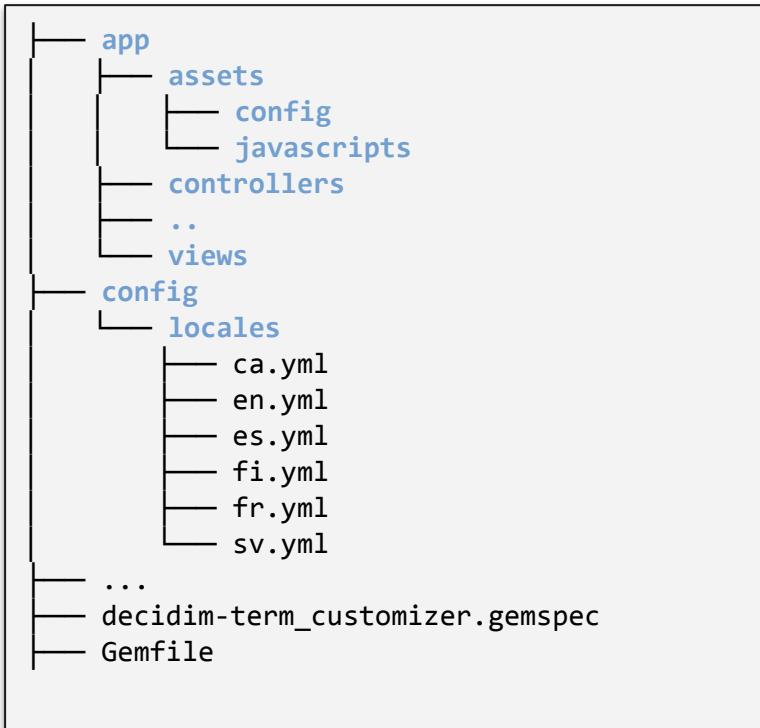
```
bin  
config  
d  
decidim-accountability  
decidim-admin  
decidim-api  
decidim_app-design  
decidim-assemblies  
decidim-blogs  
decidim-budgets  
decidim-comments  
decidim-conferences  
decidim-consultations  
decidim-core  
decidim-debates  
decidim-dev  
decidim-forms  
decidim.gemspec  
Gemfile  
...
```

<https://github.com/decidim/decidim>

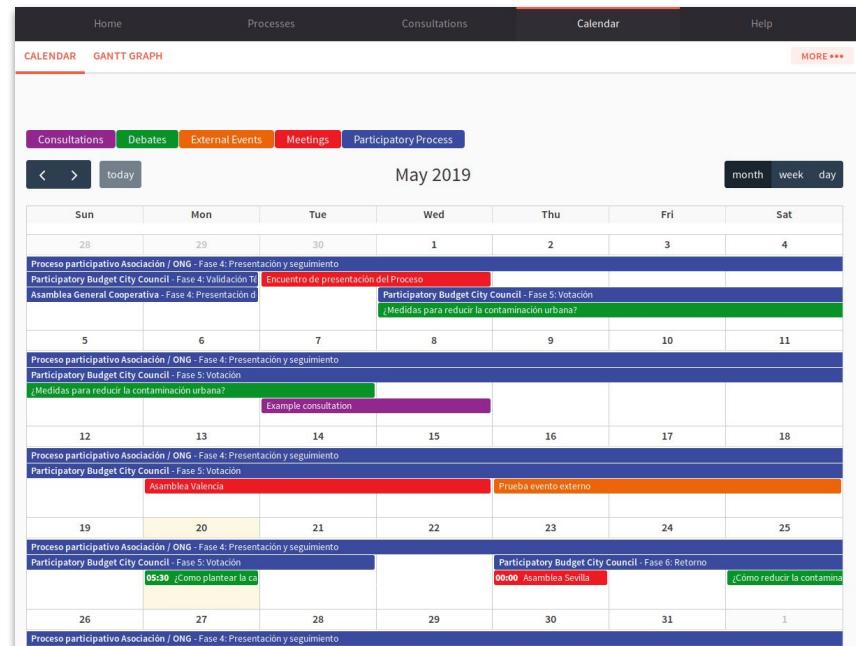


External modules

https://github.com/mainio/decidim-module-term_customizer



<https://github.com/alabs/decidim-module-calendar>



Technique 1: Overwrite Decidim HTML

- Views end in ***.erb**
- Rails looks for files in order when rendering a view (gems1 -> gem2 -> ... -> our app)
- We just need to replicate the exact path to overwrite in our **app/views** folder.
- Paths counts starting after the **app/views** part of any gem

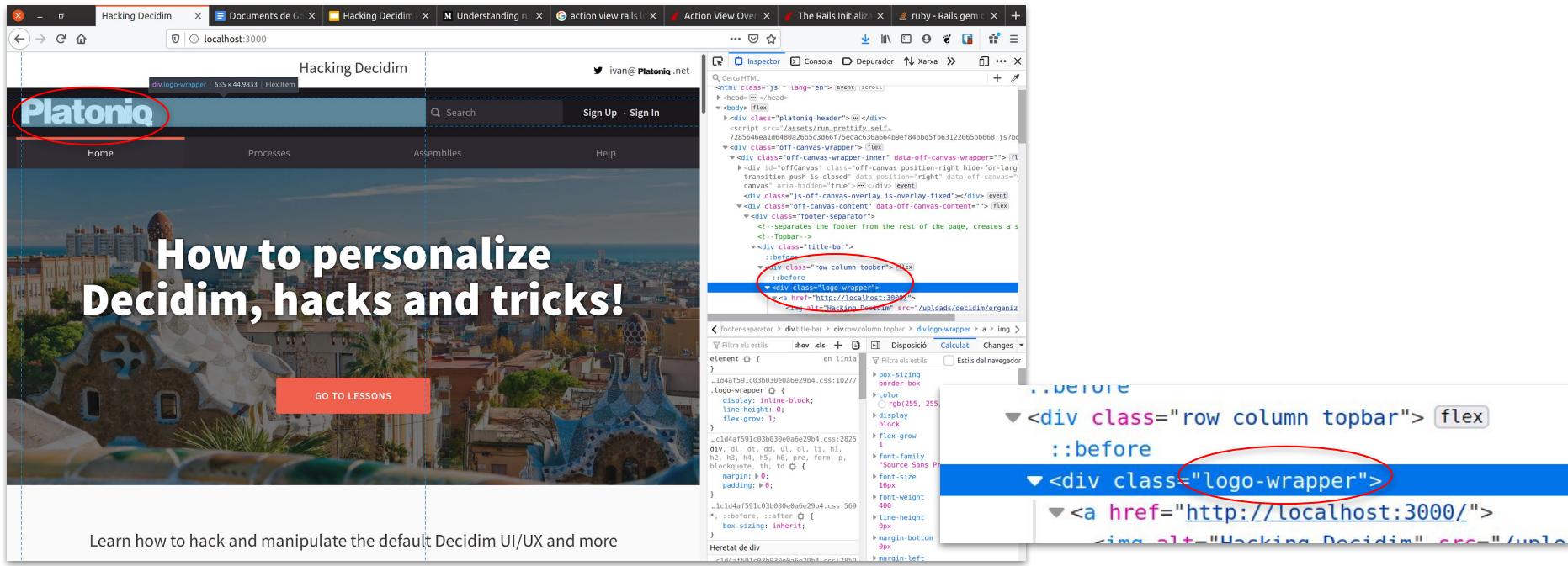
Is not always that simple:

- Some content is generated with custom classes (Cells, Presenters, Content renderers...)
- In this case we need more advanced techniques (monkey patching)

Example 1: Change something in the main layout

Step 1: Find the view to overwrite. It may be defined on any module!

TIP Use the inspector tools in your browser to locate something unique (more or less)



The screenshot shows a browser window with the URL `localhost:3000`. The page displays a header with the 'Platoniq' logo, a search bar, and 'Sign Up / Sign In' buttons. Below the header is a banner featuring a cityscape and the text 'How to personalize Decidim, hacks and tricks!'. At the bottom, there's a red button labeled 'GO TO LESSONS' and a footer with the text 'Learn how to hack and manipulate the default Decidim UI/UX and more'.

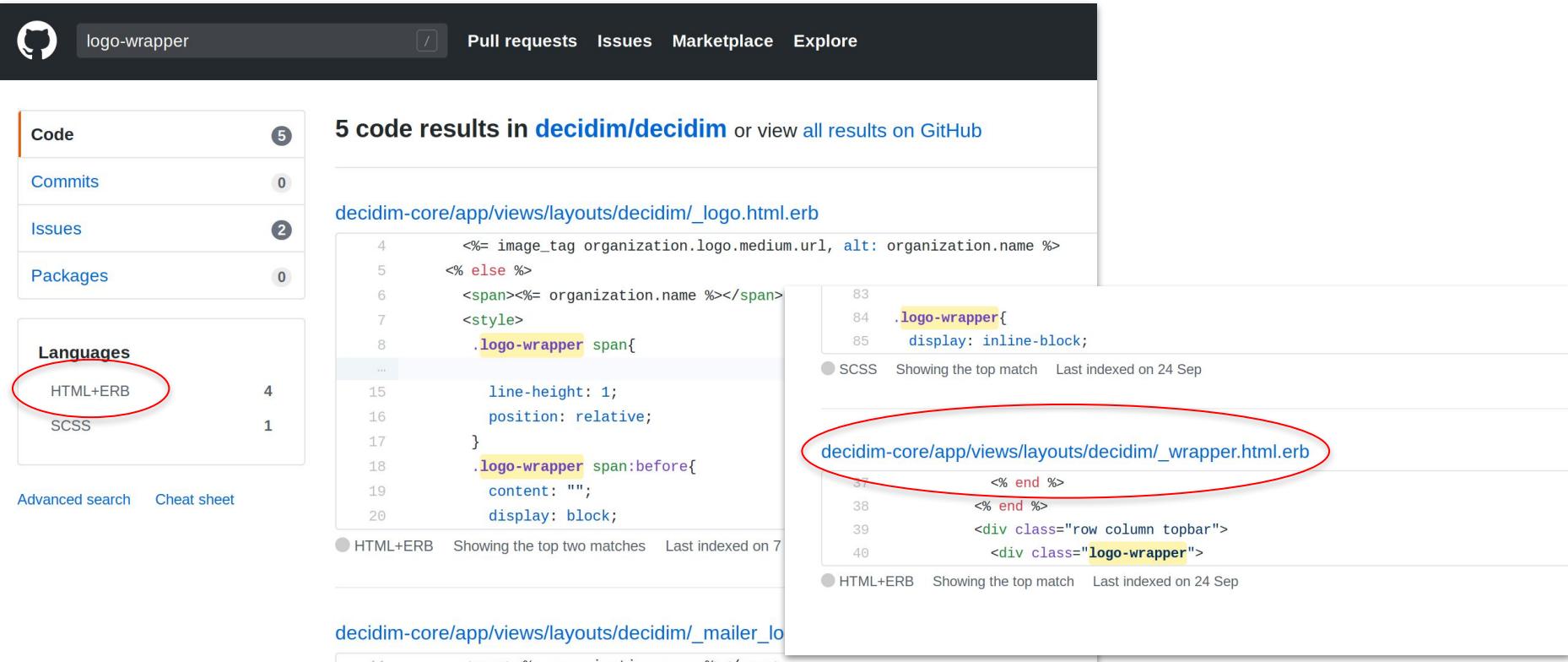
The browser's developer tools are open, specifically the 'Inspector' tab. The DOM tree shows the structure of the page, including the header, banner, and footer. A red circle highlights the 'Platoniq' logo in the header. Another red circle highlights the 'logo-wrapper' class in the CSS inspector panel, which is applied to the logo element. The CSS panel shows the styles for the 'logo-wrapper' class, including its color (rgb(255, 255, 255)) and font size (16px).

```
...BEFORE
<div class="row column topbar" flex
  ::before
    <div class="logo-wrapper">
      <a href="http://localhost:3000/">
        
      </a>
    </div>
  </div>
```

The 'Changes' tab in the developer tools is active, showing the modified CSS for the 'logo-wrapper' class. The original style was `color: #007bff; font-size: 1.25em;`, and the new style is `color: #fff; font-size: 1.25em;`.

```
...AFTER
<div class="row column topbar" flex
  ::before
    <div class="logo-wrapper">
      <a href="http://localhost:3000/">
        
      </a>
    </div>
  </div>
```

TIP Search in <https://github.com/decidim/decidim> for the located identifier



The screenshot shows the GitHub search interface for the repository `decidim/decidim`. The search term `logo-wrapper` has been entered into the search bar. The results page displays five code matches:

- decidim-core/app/views/layouts/decidim/_logo.html.erb**:
A partial ERB template containing CSS styles for a logo wrapper. The class `.logo-wrapper` is highlighted. The snippet shows the following code:

```
4      <%= image_tag organization.logo.medium.url, alt: organization.name %>
5      <% else %>
6          <span><%= organization.name %></span>
7      <style>
8          .logo-wrapper span{
```
- decidim-core/app/views/layouts/decidim/_wrapper.html.erb**:
A partial ERB template containing CSS styles for a wrapper. The class `.logo-wrapper` is highlighted. The snippet shows the following code:

```
35      <% end %>
36      <% end %>
37      <div class="row column topbar">
38          <div class="logo-wrapper">
```
- decidim-core/app/views/layouts/decidim/_mailer_lo**:
A partial ERB template starting with the same line as the first result.

The left sidebar shows repository statistics: 5 Code, 0 Commits, 2 Issues, 0 Packages, and Languages (HTML+ERB: 4, SCSS: 1). The "Languages" section is circled in red. The bottom navigation includes links for Advanced search and Cheat sheet.

PRO-TIP! Download the source code of Decidim, Open Visual Studio Code and search for that identifier locally!

Step 2:

1. Copy the file `decidim-core/app/views/layouts/decidim/_wrapper.html.erb` in your editor.
2. Change what you need.
3. Save it in your own place replicating the folder structure (`app/views/layouts/decidim/_wrapper.html.erb`)

The screenshot shows a code editor interface with the following details:

- Left Sidebar (OPEN EDITORS):** Shows 1 UNSAVED file: `_wrapper.html.erb`.
- Left Sidebar (UNTITLED (WORKSPACE)):** Shows the project structure:
 - `> decidim`
 - `< decidim-hacks`
 - `< app`
 - `> assets`
 - `> channels`
 - `> controllers`
 - `> helpers`
 - `> jobs`
 - `> mailers`
 - `> models`
 - `< views`
 - `< layouts`
 - `< decidim`
 - `< _head_extra.html.erb`
 - `< _head_platoniq.html.erb`
 - `< _wrapper.html.erb` (highlighted with a red circle)
 - `< mailer.html.erb`
- Right Panel:** Displays the content of `_wrapper.html.erb`. The code includes comments explaining the footer separation and topbar creation. A red oval highlights the section from `<div class="row column topbar">` to `</div>`, which contains a logo partial and a cool tweaks message.

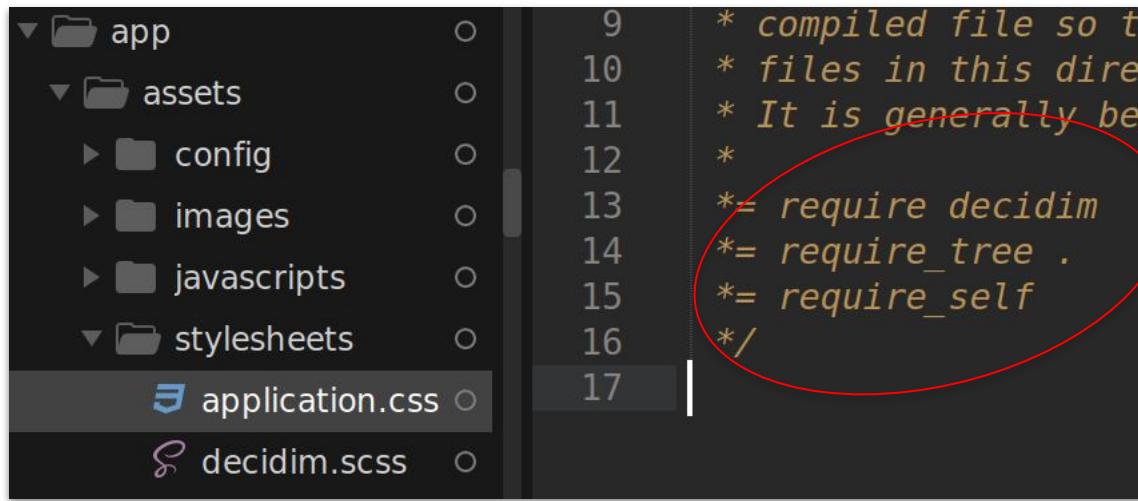
Step 3: Check in local!

The screenshot shows a web browser window with the URL `localhost:3000`. The page title is "Hacking Decidim". In the top left, there is a logo for "Platoniq" with the subtext "Cool tweaks". A red oval highlights this "Cool tweaks" link. The top navigation bar includes links for "Home", "Processes", "Assemblies", and "Help". On the right, there is a search bar, a "Sign Up" button, and a "Sign In" button. The main content area features a large image of Park Güell in Barcelona and a prominent white text overlay: "How to personalize Decidim, hacks and tricks!". Below this text is a red button labeled "GO TO LESSONS".

PRO-TIP! Now is a good time to: `git commit -a -m "add cool tweaks text"`

Technique 2: Overwrite Decidim CSS

- Most CSS is SASS formatted (***.scss**)
- Usually, we can overwrite partials the same way as HTML, the base route is now :
app/assets/stylesheets
- How CSS is loaded is defined in the main **application.css** file:



```
└── app
    ├── assets
    │   ├── config
    │   ├── images
    │   ├── javascripts
    │   └── stylesheets
    └── application.css
decidim.scss
```

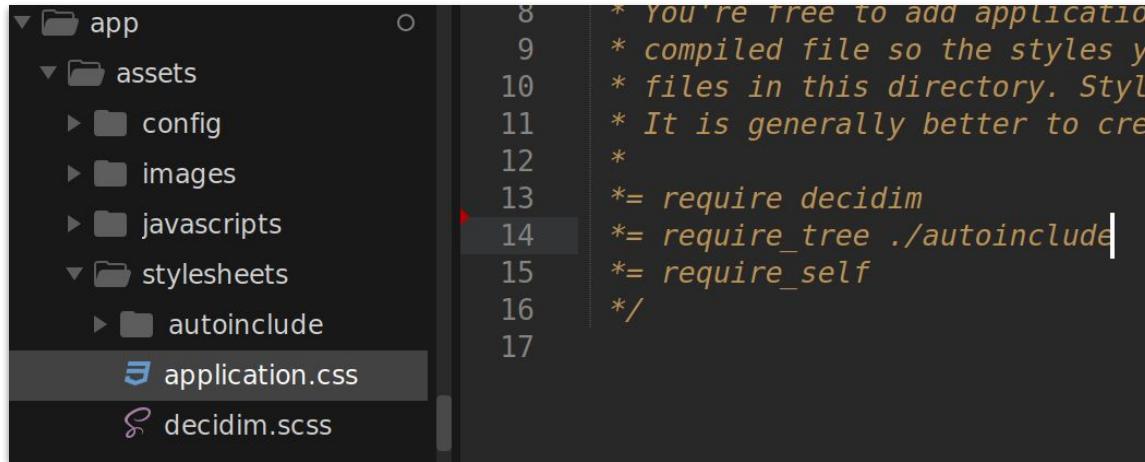
9 ** compiled file so th*
10 ** files in this direc*
11 ** It is generally bet*
12 ***
13 **= require decidim*
14 **= require_tree .*
15 **= require_self*
16 **/*
17

By default, the
require_tree .
command loads everything
we have in that folder

Technique 2: Don't mess everything!

- Some colors are changed in settings!
- Most of the time we can just put our CSS on top of whatever exists
- We want to be able to mix our overwrites with our custom CSS

TIP: Modify the `require_tree` to a custom folder. Anything we will put there will be loaded automatically.
We still have the chance to overwrite some specific view by using the duplicate path technique.



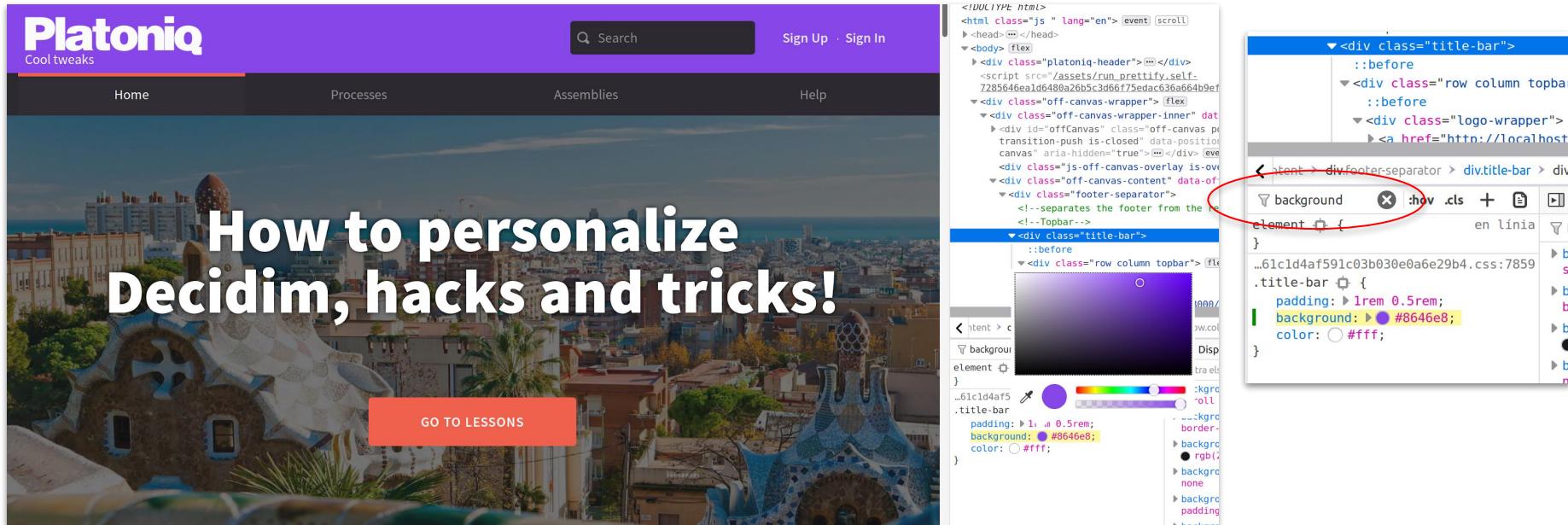
The image shows a code editor interface with a dark theme. On the left, a file tree displays the directory structure: app (with assets, config, images, javascripts, stylesheets), autoinclude, application.css, and decidim.scss. The application.css file is currently selected. On the right, the content of application.css is shown, with line numbers 8 through 17. Lines 14 and 15 contain the code `*= require_tree ./autoinclude`. The code is written in a light-colored font, and the background of the code area is dark gray.

```
8 * You're tree to add application
9 * compiled file so the styles yo
10 * files in this directory. Style
11 * It is generally better to crea
12 *
13 *= require decidim
14 *= require_tree ./autoinclude
15 *= require_self
16 */
17
```

Example 2: Change some styles in the main layout

Step 1: Find the CSS-style to overwrite. It may be defined on any module!

TIP Use the inspector tools in your browser to locate and manipulate the css class(es) implicated



The screenshot shows the Platoniq dashboard with a purple header containing the logo and navigation links for Home, Processes, Assemblies, and Help. Below the header is a banner featuring a cityscape and the text "How to personalize Decidim, hacks and tricks!". A red button at the bottom left says "GO TO LESSONS". To the right of the dashboard, a developer tools sidebar is open, showing the DOM tree, CSS styles, and a color picker. A red circle highlights the "background" style in the CSS panel, which is being edited.

Developer Tools CSS Panel:

```
.title-bar {
    padding: 1rem 0.5rem;
    background: #8646e8;
    color: #fff;
}
```

Color Picker: #8646e8

Step 2: Copy the changed CSS into a new file in our custom autoloading folder

```
padding: 0.5rem 0.5rem 1rem;  
background: #8646e8;
```

EXPLORER

> OPEN EDITORS

UNTITLED (WORKSPACE)

- > decidim
- < decidim-hacks
 - < app
 - < assets
 - > config
 - > images
 - > javascripts
 - < stylesheets
 - < autoinclude
 - < _custom_header.scss
 - < decidim-hacks.scss
 - < cool-tweaks.scss
- # application.css
- < decidim.scss

```
_wrapper.html.erb cool-tweaks.scss x  
decidim-hacks > app > assets > stylesheets > autoinclude > cool-tweaks.scss > .title-bar  
1 .title-bar {  
2   padding: 0.5rem 0.5rem 1rem;  
3   background: #8646e8;  
4 }
```

Step 3: Check in local!

Hacking Decidim

Platoniq
Cool tweaks

Search

Sign Up · Sign In

Home Processes Assemblies Help

How to personalize Decidim, hacks and tricks!

GO TO LESSONS

PRO-TIP! You're not a hacker if you don't `git commit -a -m "add 90's background"`

Technique 3: Overwrite Decidim Javascript

- You can write in traditional Javascript or ES6 (*.js and *.js.es6)
- Javascript works similar as CSS, main route is now: **app/assets/javascripts**
- How Javascript is loaded is defined in the main **application.js** file:

The terminal shows the directory structure of an application:

```
└── app
    ├── assets
    ├── config
    ├── images
    └── javascripts
        ├── channels
        ├── application.js
        ├── cable.js
        └── stylesheets
        └── controllers
```

Next to the terminal, the contents of `application.js` are displayed:

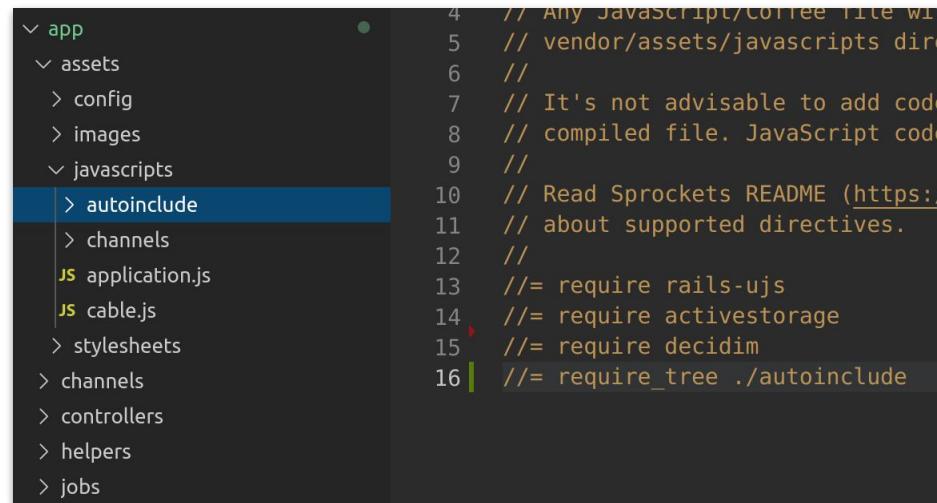
```
4 // Any JavaScript/Coffee file within
5 // vendor/assets/javascripts will be
6 // ...
7 // It's not advisable to add code
8 // compiled file. JavaScript code
9 // ...
10 // Read Sprockets README (https://github.com/rails/sprockets#readme)
11 // about supported directives.
12 // ...
13 //=- require rails-ujs
14 //=- require activestorage
15 //=- require_tree .
16 //=- require decidim
```

A red oval highlights the line `//=- require_tree .`.

By default, the `require_tree .` command loads everything we have in that folder and is located BEFORE the Decidim javascript

Technique 3: Preparation

- If we add a new behaviour, we just add some additional javascript
- If we want to modify already existing scripts, we need to overwrite that file
- If we want to make sure that any custom script is applied AFTER Decidim, we need to change the loading order (add a custom autoloader folder for the sake of organization)



The image shows a terminal window or code editor displaying a file structure and its contents. On the left, there is a tree view of files and folders under an 'app' directory. The 'autoinclude' file is selected and highlighted with a blue background. On the right, the content of the 'autoinclude' file is shown as a series of numbered lines of code.

```
4 // Any JavaScript/Coffee file with
5 // vendor/assets/javascripts dire
6 //
7 // It's not advisable to add code
8 // compiled file. JavaScript code
9 //
10 // Read Sprockets README (https://
11 // about supported directives.
12 //
13 //= require rails-ujs
14 //= require activestorage
15 //= require decidim
16 //= require_tree ./autoinclude
```

Example 3: Add some Javascript

In this example we will apply also changes in CSS and HTML.

We will change the default behaviour of the main Hero's text image by:

1. Adding an external jQuery plugin
2. Overwriting the html view to show a custom text instead of the text from admin settings
3. Add a custom JS file



Super Customizable Text Rotator with Style

PRO-TIP! Bad taste is a must for a hacker!

Step 1: Add the jQuery plugin to the system.

1. Download the jQuery plugin from:

<https://github.com/peachananr/simple-text-rotator>

- a. JS file: vendor/assets/javascripts/jquery.simple-text-rotator.js
- b. CSS file: vendor/assets/stylesheets/simpletextrotator.css

2. Edit application.css & application.js and tell them to require the downloaded files.

```
└─ stylesheets      ● 18  * You're free to add application
   └─ autoinclude    ● 19  * compiled file so the styles y
   └─ _custom_header.scss 20  * files in this directory. Styles
   └─ _decidim-hacks.scss 21  * It is generally better to create
   └─ cool-tweaks.scss   U 22  *
   # application.css    M 23  *= require decidim
                           *= require simpletextrotator
                           *= require_tree ./autoinclude
                           *= require_self
   └─ decidim.scss      24  */
   └─ decidim.scss      25
   └─ decidim.scss      26
   └─ decidim.scss      27  */
```

```
└─ javascripts      ● 13 // Read Sprockets README (https://github.com/rails/sprockets)
   └─ autoinclude      ● 14 // about supported directives.
   └─ hero-fx.js       U 15 //
   > channels
   └─ application.js   M 16 //= require rails-ujs
                           //= require activestorage
                           //= require decidim
                           //= require jquery.simple-text-rotator
   └─ cable.js          17
                           18
                           19
                           20
```

IMPORTANT! RELOAD THE SERVER NOW:

```
~ $ docker-compose exec app bin/rails restart
```

Step 2: Find the view to overwrite.

TIP Use the inspector tools in your browser to locate a useful identifier (ie: hero-heading)

The screenshot shows the Platoniq homepage with a purple header containing the logo, search bar, and sign-in links. Below the header is a large hero section featuring a cityscape background and a central text area. A red dashed box highlights the text "How to personalize Decidim, hacks and tricks!". A callout box with the text "h1.text-highlight.heading1.hero-heading | 876.667 x 144" points to the first heading in the text area. To the right, the browser's developer tools are open, specifically the element inspector. A red circle highlights the "hero-heading" class in the CSS panel, which is applied to the first

element in the DOM tree. The DOM tree shows the structure of the page, including sections like "extended hero home-section" and "extended subhero home-section". The bottom of the developer tools shows the element and style panels.

```
<!-->
</div>
<div class="show-for-medium" data-set="nav-holder">...</div>
<main>
  <section class="extended hero home-section" style="background-image:url(/uploads/decidim/hero_homepage_content_block/background/1/big_homepage_image.jpg);">
    <div class="hero__container">
      <div class="row">
        <div class="columns small-centered large-10">
          <h1 class="text-highlight heading1 hero-heading">...</h1>
        </div>
      </div>
    </div>
  </section>
  <section class="extended subhero home-section">...</section>
  <section class="wrapper-home home-section">...</section>
  <section class="wrapper-home home-section">...</section>
```

element {
 en línia
 □ Filtra els estils
 □ Estils del navegador
}

background
x show cls + □ Disposició Calculat Changes

box-sizing border-box
color
○rgb(255, 255, 255)
font-family "Source Sans Pro", Helvetica, R...
font-size 72px

hero-heading seems reasonably unique

NOTE: It is not a regular **html.erb** file!

The screenshot shows a GitHub search interface. The search bar at the top contains the query "hero-heading". Below the search bar, the navigation menu includes Pull requests, Issues, Marketplace, and Explore. On the left, there are two sections: "Code" (8 results) and "Languages" (HTML+ERB 7, SCSS 1). The main content area displays search results for "hero-heading". A red oval highlights the first result, which is a file named "decidim-core/app/cells/decidim/content_blocks/hero/show.erb". The code snippet from this file is shown:

```
4   <div class="columns small-centered large-10">
5     <h1 class="text-highlight heading1 hero-heading">
6       <% if translated_welcome_text.blank? %>
```

Below the code snippet, it says "HTML+ERB Showing the top match Last indexed on 30 Aug 2018". The second result listed is "decidim_app-design/app/views/public/partials/_conferences_hero.html.erb", with the following code snippet:

```
5   <div class="columns large-10 text-left">
6     <h1 class="text-highlight hero-heading mb-none">
7       IV Jornades MetaDecidim
```

Below this, it says "HTML+ERB Showing the top match Last indexed on 10 Sep 2018".

Step 3: Overwrite the original view by creating a new one.

1. Replicate folder structure and copy the original file:

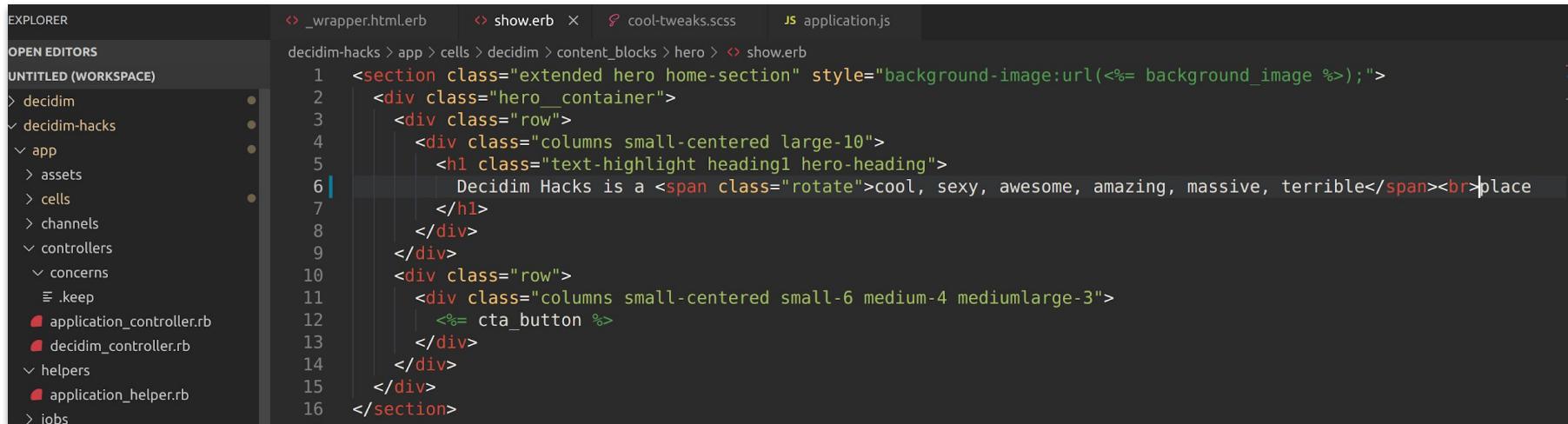
`decidim-core/app/cells/decidim/content_blocks/hero/show.erb`



2. In our folder: `app/cells/decidim/content_blocks/hero/show.erb`

Step 4: Edit the file. app/cells/decidim/content_blocks/hero/show.erb

We will remove any string coming from the database (sorry admins, no more Hero editing for you) and put a fixed phrase:



The screenshot shows a code editor interface with the following details:

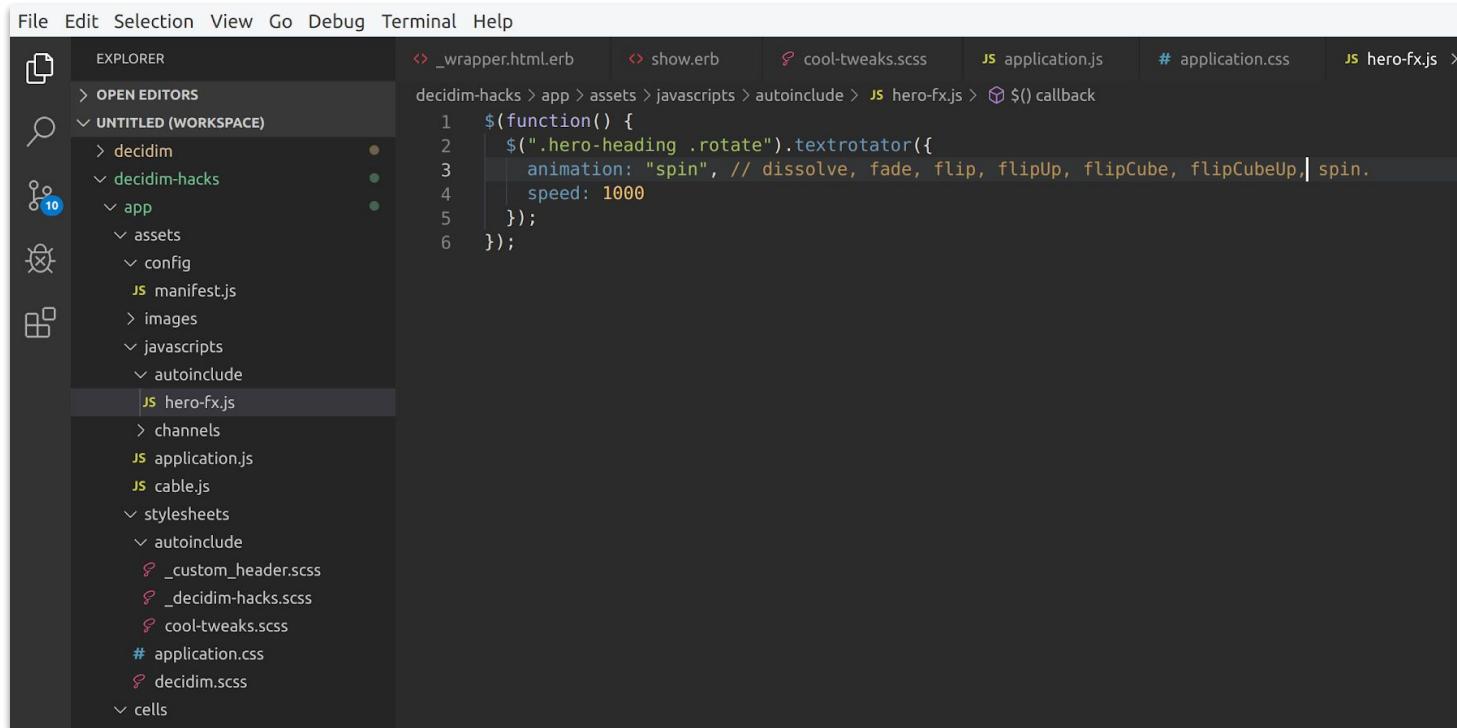
- EXPLORER** sidebar: Shows the project structure under "decidim-hacks".
- OPEN EDITORS** tab bar: Includes files like _wrapper.html.erb, show.erb (which is currently selected), cool-tweaks.scss, and application.js.
- Content Area:** The file content is as follows:

```
decidim-hacks > app > cells > decidim > content_blocks > hero > show.erb
1  <section class="extended hero home-section" style="background-image:url(<%= background_image %>);">
2    <div class="hero__container">
3      <div class="row">
4        <div class="columns small-centered large-10">
5          <h1 class="text-highlight heading1 hero-heading">
6            Decidim Hacks is a <span class="rotate">cool, sexy, awesome, amazing, massive, terrible</span><br>place
7          </h1>
8        </div>
9      </div>
10     <div class="row">
11       <div class="columns small-centered small-6 medium-4 mediumlarge-3">
12         <%= cta_button %>
13       </div>
14     </div>
15   </div>
16 </section>
```

Decidim Hacks is a cool, sexy, awesome, amazing, massive, terrible
place

Step 5: Create your own javascript that initializes everything, for instance:

app/assets/javascripts/autoinclude/hero-fx.js

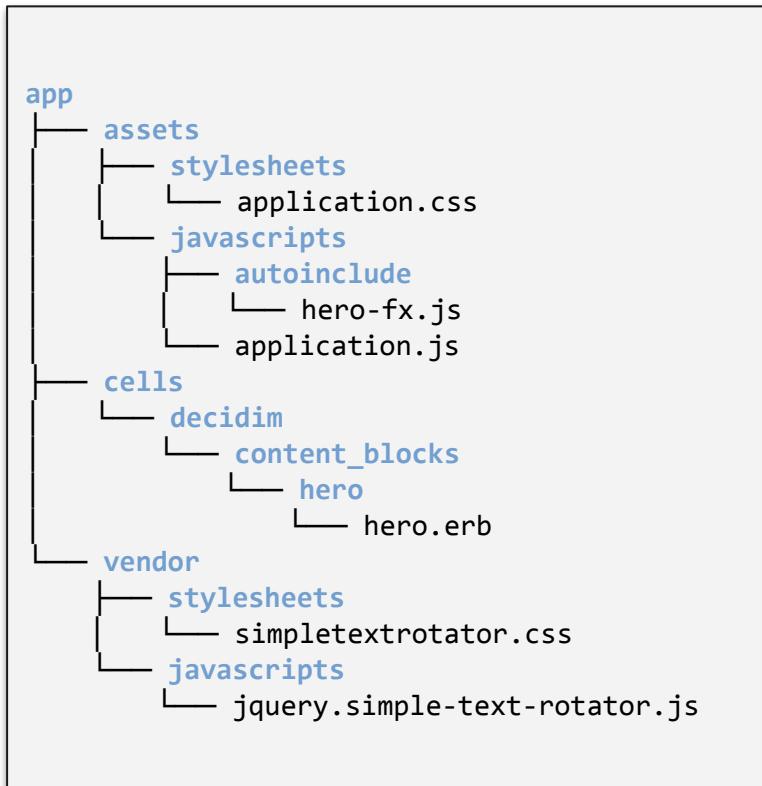


The screenshot shows a code editor interface with a dark theme. The top menu bar includes File, Edit, Selection, View, Go, Debug, Terminal, and Help. The left sidebar displays a file tree under 'UNTITLED (WORKSPACE)'. The 'autoinclude' folder contains files like '_wrapper.html.erb', 'show.erb', 'cool-tweaks.scss', 'application.js', 'application.css', and 'hero-fx.js'. The 'hero-fx.js' file is currently selected and open in the main editor area. The code in 'hero-fx.js' is as follows:

```
$(function() {
  $(".hero-heading .rotate").textrotator({
    animation: "spin", // dissolve, fade, flip, flipUp, flipCube, flipCubeUp, spin.
    speed: 1000
  });
});
```

Recapitulation:

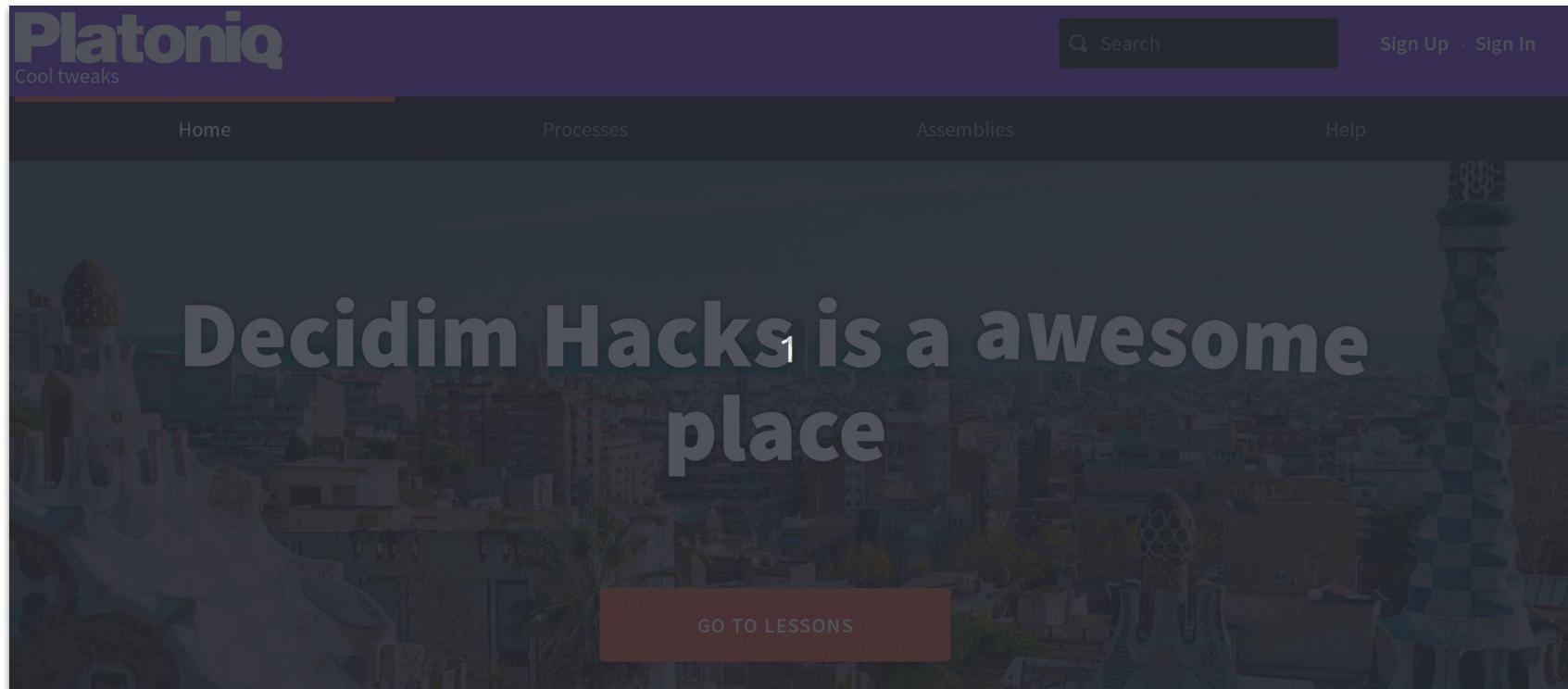
Files changed:



Steps:

1. Added external (vendor) javascript/css to the project
2. Configured Rails to load those files
3. Restarted Rails
4. Overwritten the Hero view.
5. Applied some custom javascript (activate the plugin)

Step 6: Enjoy!



Technique 4: Overwrite locales

- Locales are the multi-language systems texts
- Written in YAML format.
- Inside **app/config/locales/*.yml** (every module/gem has its own).
- To overwrite, just create the same key with different content (for each language active!)

The screenshot shows the GitHub search interface with the query "Propostes". The results are filtered by "YAML" and show two matches:

- decidim-proposals/config/locales/ca.yml**

```
29     copy_proposals: Entenc que això importarà totes les propostes del
        component seleccionat a l'actual i que aquesta acció no es pot desfer.
30     origin_component_id: Component des d'on importar les propostes
```
- decidim-sortitions/config/locales/ca.yml**

```
5         additional_info: Informació del sorteig
6         decidim_category_id: Categories del conjunt de propostes en què vols
        aplicar el sorteig
7         decidim_proposals_component_id: Propostes establertes
8         dice: Resultat de la tirada de daus. Tira un dau de sis cares o busca
        alguna altra manera de generar un número de l'u al sis i introduceix-lo aquí
        davant d'algunes testimonies. Això contribueix en la qualitat i garantia de
        l'aleatorietat del resultat
```

Below the results, there are links for "Advanced search" and "Cheat sheet".

Example 4: Add custom locales

In this example we will create a new FX like the previous example to use multi-language texts.

We will:

1. Create the list of words that rotate as independent text entries in the app locales folder.
2. Create a “helper” ruby method that will list the words for us in the view.
3. Add a new javascript file to take care of this case.

NOTE: Helpers cannot be defined this way in the previous example (it is not a regular rails **html.erb** file).

Step 1: Edit the locales file: config/locales/en.yml

Add some content in the YAML format:

The image shows a code editor interface with a dark theme. On the left, a tree view displays the project structure:

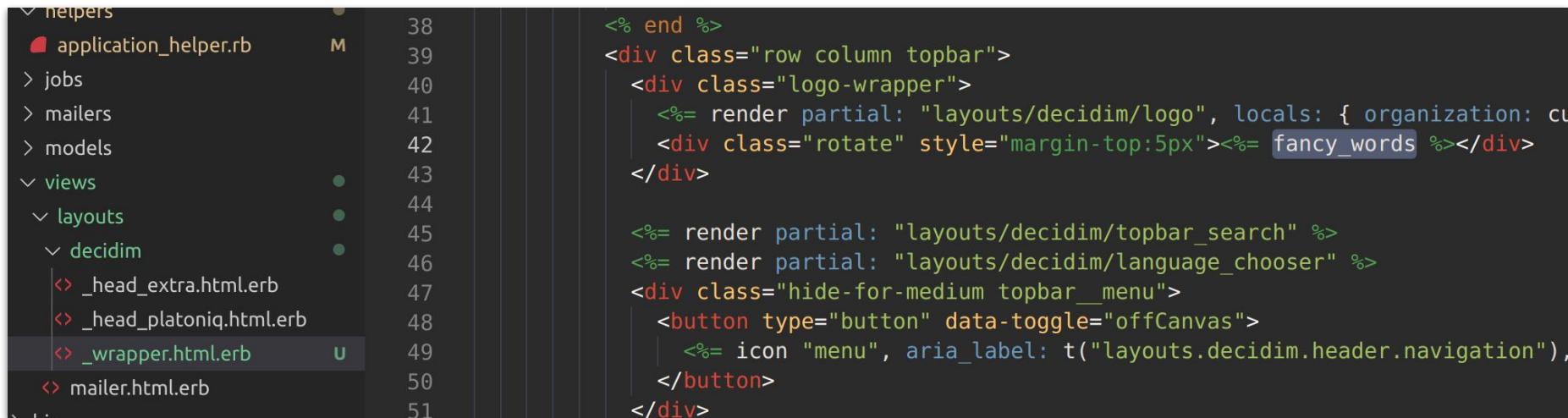
- _head_platoniq.html.erb
- _wrapper.html.erb
- mailer.html.erb
- bin
- config
 - environments
 - initializers
- locales
 - en.yml
- application.rb
- boot.rb
- cable.yml

The file `en.yml` is selected and highlighted with a blue bar at the bottom. The right pane shows the YAML content:

```
29 # To learn more, please re
30 # available at http://guide
31
32 en:
33   hello: "Hello world"
34   hacker_words:
35     word1: Cool tweaks
36     word2: Sexy tweaks
37     word3: Awesome tweaks
38     word4: Amazing tweaks
39     word5: Massive tweaks
40     word6: That's ok
41     word7: I get it...
```

Step 2: Edit the layout file: app/views/layouts/decidim/_wrapper.html.erb

Change the previous "Cool tweaks" for the method we are going to use:



The screenshot shows a code editor with a dark theme. On the left, there's a tree view of files under the 'decidim' directory. The 'views/layouts/decidim/_wrapper.html.erb' file is selected and highlighted with a green background. The right pane displays the content of this file.

```
<% end %>
<div class="row column topbar">
  <div class="logo-wrapper">
    <%= render partial: "layouts/decidim/logo", locals: { organization: cu
    <div class="rotate" style="margin-top:5px"><%= fancy_words %></div>
  </div>

  <%= render partial: "layouts/decidim/topbar_search" %>
  <%= render partial: "layouts/decidim/language_chooser" %>
  <div class="hide-for-medium topbar_menu">
    <button type="button" data-toggle="offCanvas">
      <%= icon "menu", aria_label: t("layouts.decidim.header.navigation"), %>
    </button>
  </div>
```

Step 3: Edit the helper's file: app/helpers/application_helpers.rb

Create the method fancy_words:

The screenshot shows a code editor interface with the following details:

- EXPLORER** panel on the left showing the project structure:
 - decidim-hacks > app > helpers > application_helper.rb
 - controllers
 - concerns
 - .keep
 - application_controller.rb
 - decidim_controller.rb
 - helpers > application_helper.rb
 - jobs
- OPEN EDITORS** panel showing the current file: application_helper.rb.
- application_helper.rb** content pane showing the following Ruby code:

```
1 module ApplicationHelper
2   def fancy_words
3     words = (1..7).map do |number|
4       I18n.t("hacker_words.word#{number}")
5     end
6     words.join(', ')
7   end
8 end
9
```

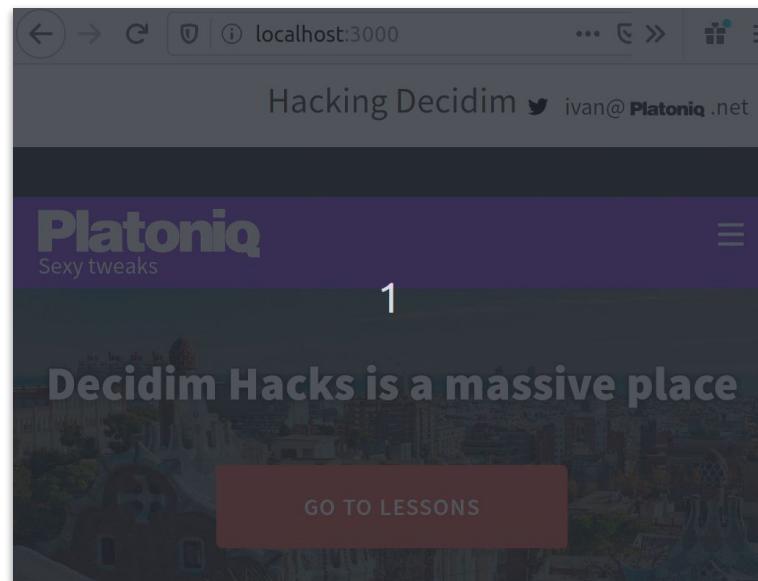
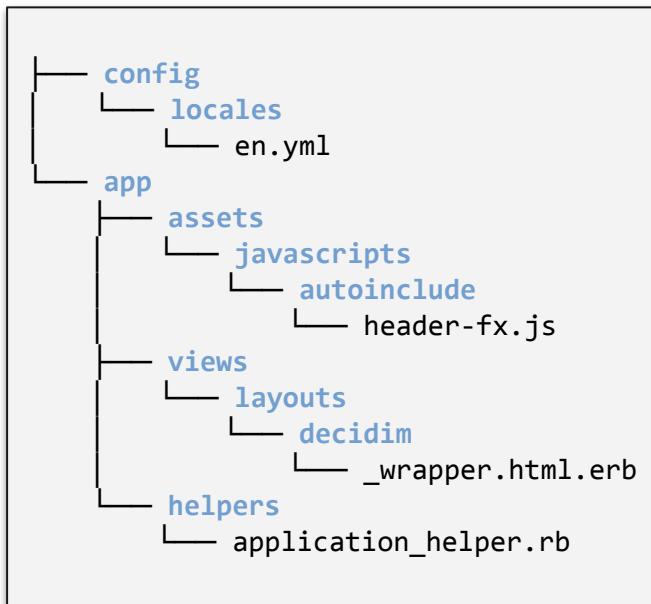
Step 4: Create the javascript associated file to initialize the plugin:

app/assets/javascripts/autoinclude/header-fx.js:

The screenshot shows a code editor interface with a dark theme. On the left, there's an 'EXPLORER' sidebar showing a file tree under 'UNTITLED (WORKSPACE)'. The 'autoinclude' folder contains 'header-fx.js' and 'hero-fx.js'. The 'header-fx.js' file is open in the main editor area. The status bar at the bottom indicates '1 UNSAVED' changes.

```
<> show.erb      <> _wrapper.html.erb      <> _head_extra.html.erb      JS header-fx.js ●      application_helper.rb
decidim-hacks > app > assets > javascripts > autoinclude > JS header-fx.js > ...
1  $(function() {
2    $(".logo-wrapper .rotate").textrotator({
3      animation: "flip", // dissolve, fade, flip, flipUp, flipCube, flipCubeUp, spin,
4      speed: 1000
5    });
6  });
7
```

Step 5: Check and recapitulate



PRO-TIP! Hackers are humans too... `git commit -a -m "😱 please no more effects..."`

Technique 5: Monkey Patching

Metaprogramming

- Ruby is an extremely manipulable programming language
- Almost everything can be redefined.
- We will use this capability to redefine methods defined in the Decidim core without having to touch it

Hooks

- Ruby (as a language) has hooks when certain events happen
- `class_eval` for creating instance methods
- `instance_eval` for creating class methods



What is monkey patching?

From the Decidim gem, loaded first

```
1 Decidim
2   def say_hello
3     "hello"
4   end
5 end
```

In our app, loaded after Decidim

```
1 Decidim
2   def say_hello
3     "bye"
4   end
5 end
```

Where to monkey patch in Decidim

PROBLEM: Most of the classes are already instantiated when Rails load our app.

- We will use Ruby hooks to append our code in a custom Rails initializer, usually `class_eval`
- A Rails initializer is just a Ruby file inside `config/initializers/something.rb`. It's evaluated when the application starts (once only)
- Changing a initializer requires Rails reload

Example 5: Overwrite non-configurable settings

In this example we will monkey-patch the Proposals method that check the titles of the proposals

We will:

1. Identify the Proposal validation method
2. Create a new initializer and overwrite the method to allow more flexible titles in proposals

The screenshot shows a user interface for creating a proposal. On the left, there's a sidebar with navigation options: 'Create your proposal' (highlighted in red), 'Compare', 'Complete', and 'Publish your proposal'. The main area has a header 'CREATE YOUR PROPOSAL'. It contains two input fields: 'Title *' with the value 'i am short' and an error message 'must start with a capital letter, is too short (under 15 characters)'; and 'Body *' with the value 'Hello, I want to hack!'. Below these is a dropdown 'Create proposal as' set to 'Eve O'Hara'. At the bottom is a large orange 'Continue' button.

Step 1: Find the class with the method.

This can be difficult and requires time to study the code to do it properly. But... we're feeling brave!

TIP: start to search for the locale key and then where is it placed in the code:

A screenshot of a GitHub search interface. The search bar at the top contains the query "must start with a capital". To the right of the search bar is a "Search" button. Below the search bar, there is a sidebar with navigation links: "Code" (1 result), "Commits" (0), "Issues" (1), and "Packages" (0). Under "Languages", "YAML" is selected, indicated by a grey background, and shows 1 result. The main content area displays the search results for "decidim-core/config/locales/en.yml". The file contains YAML configuration for a "start" command. Line 1225 shows the key "must_start_with_caps" with its value "must start with a capital letter" highlighted in yellow. A red oval highlights the file path "decidim-core/config/locales/en.yml". A red arrow points from the search term in the GitHub search bar to the highlighted file path in the results.

must start with a capital

Code 1

Commits 0

Issues 1

Packages 0

Languages YAML 1

decidim-core/config/locales/en.yml

```
title: Reply
show:
  title: Conversation with %{usernames}
start:
  send: Send
...
long_words: contains words that are too long (over 35 characters)
must_start_with_caps: must start with a capital letter
nesting_too_deep: can't be inside of a subcategory
```

A screenshot of a user interface showing a validation error. The input field is labeled "Title *". It contains the text "i am short", which is underlined in red. Below the input field, an error message is displayed in red: "must start with a capital letter, is too short (under 15 characters)". A red oval encircles both the error message and the input field.

Title *

i am short

must start with a capital letter, is too short (under 15 characters)

Search where the locale key is used

Search

| | |
|----------|---|
| Code | 1 |
| Commits | 0 |
| Issues | 0 |
| Packages | 0 |

| | |
|-----------|----|
| Languages | 25 |
| Ruby | X |

1 code result in **decidim/decidim**

Bingo!

[decidim-core/app/validators/etiquette_validator.rb](#)

```
39     record.errors.add(attribute, options[:message] || :must_start_with_caps)
40   end
41
42   def validate_length(record, attribute, value)
```

● Ruby Showing the top match Last indexed on 18 Jun

Step 2: Create a new initializer file. It will be used to monkey-patch the method found.

For instance:

config/initializers/bananas_for_proposals.rb

Also: open the original file to copy and analize what we need.

```
# This validator takes care of ensuring the validated content is
# respectful, doesn't use caps, and overall is meaningful.

class EtiquetteValidator < ActiveModel::EachValidator
```

```
def validate_length(record, attribute, value)
  return if value.length > 15

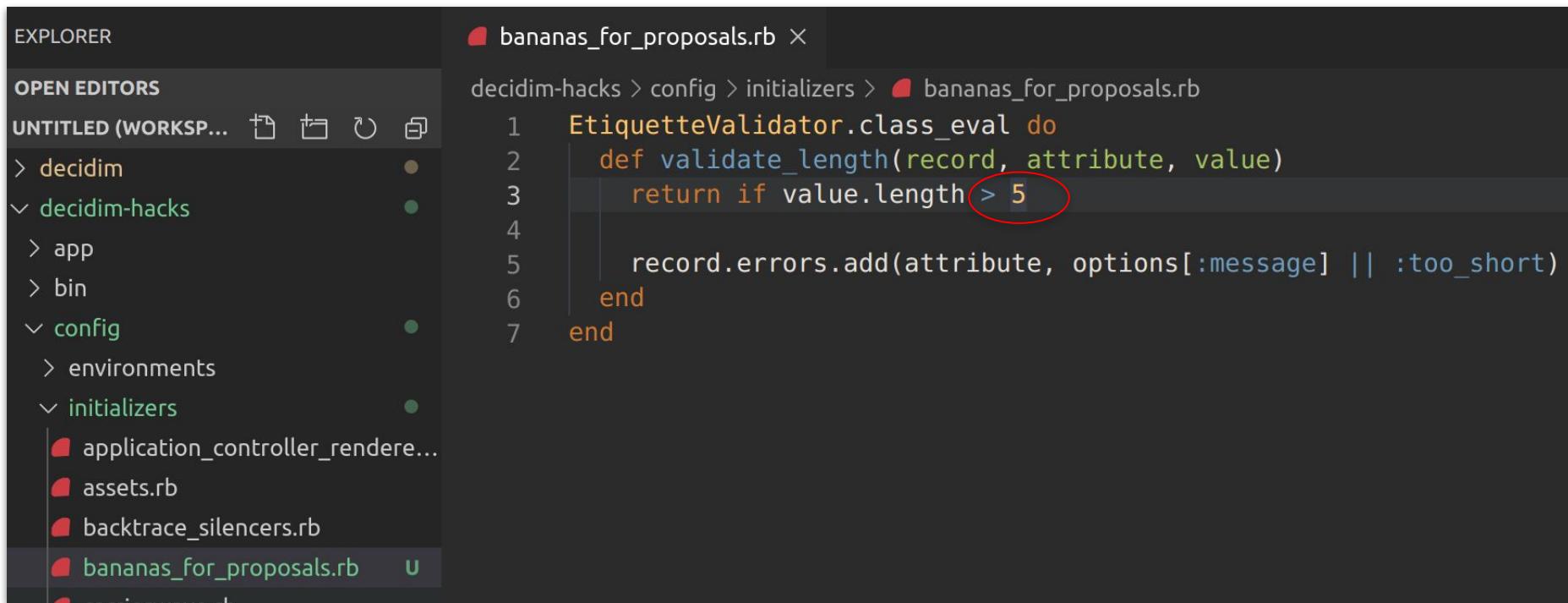
  record.errors.add(attribute, options[:message] || :too_short)
end
```

https://github.com/decidim/decidim/blob/cc44574a14f7935dc6195de0c43486f69d824830/decidim/app/validators/etiquette_validator.rb

```
2
3   # This validator takes care of ensuring the validated content is
4   # respectful, doesn't use caps, and overall is meaningful.
5   class EtiquetteValidator < ActiveModel::EachValidator
6     def validate_each(record, attribute, value)
7       return if value.blank?
8
9       validate_caps(record, attribute, value)
10      validate_marks(record, attribute, value)
11      validate_long_words(record, attribute, value)
12      validate_caps_first(record, attribute, value)
13      validate_length(record, attribute, value)
14    end
15
16    private
17
18    def validate_caps(record, attribute, value)
19      return if value.scan(/[A-Z]/).length < value.length / 4
20
21      record.errors.add(attribute, options[:message] || :too_much_caps)
22    end
23
24    def validate_marks(record, attribute, value)
25      return if value.scan(/![?;]{2,}/).empty?
26
27      record.errors.add(attribute, options[:message] || :too_many_marks)
28    end
29
30    def validate_long_words(record, attribute, value)
31      return if value.scan(/\w{35,}/).empty?
32
33      record.errors.add(attribute, options[:message] || :long_words)
34    end
35
36    def validate_caps_first(record, attribute, value)
37      return if value.scan(/^A[a-z]{1}/).empty?
38
39      record.errors.add(attribute, options[:message] || :must_start_with_caps)
40    end
```

Step 3: Edit the file and register our hook for the class we want to monkeypatch.

Copy the method exactly as it is. Then modify what you want.



The screenshot shows the Visual Studio Code interface. On the left, the Explorer sidebar displays the project structure under 'OPEN EDITORS'. The 'config/initializers' folder contains several files, with 'bananas_for_proposals.rb' currently open. The code editor shows the following Ruby code:

```
EtiquetteValidator.class_eval do
  def validate_length(record, attribute, value)
    return if value.length > 5
    record.errors.add(attribute, options[:message] || :too_short)
  end
end
```

The line 'return if value.length > 5' is circled in red, indicating it is the part to be modified. The status bar at the bottom right shows the file name 'bananas_for_proposals.rb' and a small 'U' icon.

Step 4: Reload the server. Everything modified inside initializers requires a server reload.

◀ Back

CREATE YOUR PROPOSAL

You are creating a proposal.

Create your proposal

Compare

Title *

i am short

must start with a capital letter

Body *

✗ Behaviour modified! 🍻

◀ Back

COMPLETE YOUR PROPOSAL

You are creating a proposal.

Create your proposal

Compare

Complete

Title *

I am short

Body *

Hello, I want to hack!

Inspire yourself: real examples

<https://omastadi.hel.fi/>

<https://plazapublica.cdmx.gob.mx/>

<https://dddc.decodeproject.eu/>

<https://labs.openheritage.eu/>



<https://github.com/City-of-Helsinki/decidim-helsinki>

<https://github.com/ponentesincausa/PlazaPublicaCDMX>

<https://github.com/DECODEproject/DDDC-instance>

<https://github.com/Platoniq/decidim-openheritage>



<https://github.com/decidim/decidim/network/dependents>

Conclusions

1. Don't do it!
2. Be aware of upgrades!
3. Touch the minimum possible.
4. Test extensively in local before deploying in production



Leave your comments in

<https://github.com/Platoniq/decidim-hacks/issues>

or just write me:

ivan@platoniq.net



<https://twitter.com/ivanverges>



<https://github.com/microstudi>



<https://meta.decidim.org/profiles/microstudi>



ivan@platoniq.net