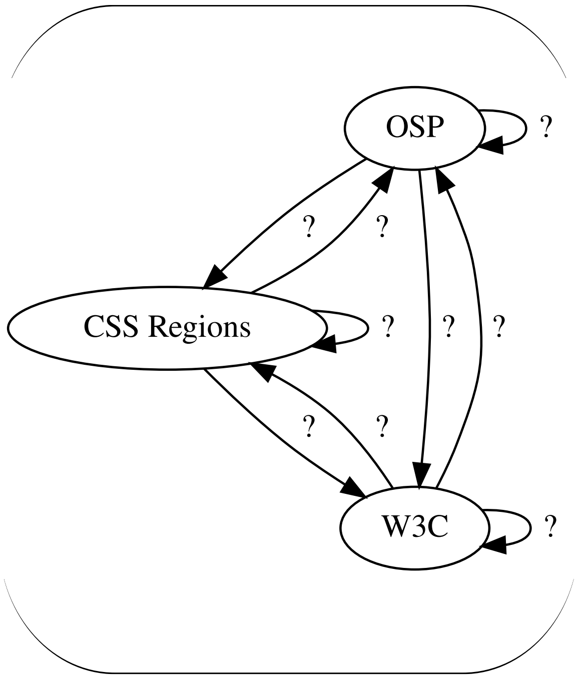
CSS Regions: dancing in the ruins of the web

a multi-centered story of 10 years of web-to-print practices based on a never fully implemented CSS standard



SWITCHING TO "PUBLISHING PARTYLINE BROADCASTS: STANDARDS AND WORKAROUNDS": A RADIO INTERVIEW WITH OSP ON 7 SEPTEMBER 2022, WHICH WAS A FIRST MOMENT TO OPEN UP THE CONVERSATION AROUND THE CSS REGIONS.

Simon Browne and i hosted this interview together in preperation of the Publishing Partyline, a two-day gathering in Varia around web-to-print practices in October 2022. The interview was broadcast live from the OSP studio on Varia's Narrowcast, you can listen back to it here: https://cc.vvvvvvaria.org/wiki/Standards_and_work_arounds. Simon and i were joined by Amélie Dumont, Gijs de Heij, Alex Leray and Doriane Timmermans from OSP.

At the time of this conversation i had heard of the CSS Regions stories, but did not know any of the details yet. At a certain point...

Manetta: Can you maybe explain what CSS Regions are and how it works?

Doriane: Yeah. [laughter] So CSS Regions is mainly a set of CSS properties. And the way it works, is that's it separates a bit the content from the layout, in the sense that you still have all your `<div>`'s and paragraphs in one content `<div>`, but then you're able to let all your content flow in another set of `<div>`'s. Which are basically kind of empty `<div>`'s, because when you inspect them there is not the real content in them, but the browser renders it as such that the text is inside of it. So what it allows you to do is that you have one big flow of content, and todiv ide it into seperate layout flows, and to place each of these flows into a different `<div>`. So it's helpful to make magazine layouts and printed media in general.

Manetta: How important have the CSS Regions been for OSP?

Alex: I think the first reason to use CSS Regions was to do a multi-paged document. Because if you have a printed booklet, it might be as simple as you want, like one column of text. But you might have a footer, or you might have an image on the right page, and then you want to continue the text on the page after. So at some point it was kind of the solution to that, within this kind of broken environment of web-to-print at the time. So it was not so much... because then it's funny to say where the CSS Regions came in, but was not so much about... It was a little bit like problem solving for this multi-paged output. That's the way that we found to do more fragmented layouts, and also to go away a bit from the linearity from the web. But it also was at some costs in a way.

Manetta: Yes, let's jump to the costs directly. So in 2013 big changes were made to the Chrome and Chromium browsers: Google, the maker of these browsers, decided to switch to a different browser engine. Google forked Apple's browser engine WebKit and started Blink, a new browser engine. And as part of this change, they also decided to remove the support for CSS Regions.

Maybe we should start with explaining what a browser engine is, before we continue? Because that is quite important.

Gijs: So a browser engine is a piece of software that translates the code of a web page into pixels, or the code into an image. So it combines the content together with the rules for the layout and the constraints that are there, for example the width of your browser window, and then calculates or determines what the layout should be.

Maybe a clear example is to think about an image with a width of 50% and a text flowing next to it. If your screen is very wide, the image will become bigger, but also more text fits next to it. So that needs to be calculated. And if your screen is smaller, then the image is smaller as well and the text has to flow differently.

So that's what this engine does. It takes the instructions or the limitations set in CSS and combines it with the content that it finds in the HTML, and it determines what is looks like on your screen.

Manetta: And you could work with CSS Regions because they were implemented in the WebKit browser engine, right? Can you say a bit more about WebKit? What made you aware that you were reyling on this particular browser engine?

Alex: First Chrome was running on Blink...

Gijs: No WebKit.

Alex: On WebKit, sorry. And they were sharing the same web rendering...

Gijs: Engine.

Alex: ...engine –thank you– with Safari basically. And Chrome took some shares on the market. At some point they decided that they wanted to continue the development of the browser, they probably disagreed with something, I don't know the story, but I think there was some kind of disagreement.

Gijs: I think, in my mind, CSS Regions was the reason for the split. In the sense that there were blog posts about the enormity of... Let's say, there were a lot of lines of code that were there specifically to support CSS Regions. And the developers wanted to decrease the size of Blink.

And also, which is something else, the CSS Regions have initially been proposed as a standard by Adobe. It very closely imitates the idea that Adobe has about layout, where you draw boxes on a page and there's content flowing into the boxes, very much like how their tool InDesign works. And there's also kind of a clear relationship between Adobe and Apple. As in, I think at that moment, the most important reason for people to use Apple computers was because Adobe software was running on it. So I also think that that heavily influenced Adobe's proposal and their interest in the WebKit project.

And Google wanted to remove CSS Regions, or at least that is my understanding of the story. They wanted to remove the CSS Regions functionality, because it would make the browser faster.

Manetta: Yes that is what we also read. That CSS Regions occupied 10.000 lines of code, which could be removed from the browser engine basically, which was written in 350.0000 lines of C++ code in total.

Let's go back to OSP... did you heavily rely on Chrome in your practice?

Alex: I think when we discovered CSS Regions, I think we used Chromium. Which is an open source... it is a version of the Chrome browser on Linux. But we used it only for a very brief time, if I remember it correctly, because right after Chrome and thus also Chromium decided to remove the CSS Regions functionality.

Gijs: Safari does not run on Linux. So at that moment Chromium was the biggest browser on the Linux platform that used the WebKit rendering engine.

Manetta: Just to clarify, you all the using Linux in your practice? That is an important detail.

Together: Yes.

Manetta: So the browser you were using to produce your work in, stopped supporting the CSS Regions.

Alex: Exactly.

Manetta: Which meant that the way in which you were producing layouts with HTML and CSS was not working anymore, thanks to switch of Chrome from WebKit to Blink in 2013? That must have been quite scary. How did you respond to it?

Alex: I think we, we tried..., I mean... we started a bit panicking I think. Not because we liked so much this CSS Regions functionality, because like I said, it was our only way at the time, or the only way how you could think about multi-page layout in the web browser. And we were not so much enthusiastic to come back to the former tools, such as Scribus. We liked working with the web so much that we wanted to continue like that, even though we had some reservations about CSS Regions itself.

Alex: What we tried was to use a polyfill, that was developed by a student at Adobe, actually, to circumvent or to re-implement in a way this idea of CSS Regions.

What we found was that it was very nice to have this code around, but it was also very difficult to work with the Javascript implementations of it. Because first of all, it was written in Javascript which is not a low level programming language and it made it very very slow when working on large documents. And second, it was breaking some nice CSS features, like selectors, which you use for instance if you want to select the first paragraph of your document. And when using the polyfill, it will suddenly select the first paragraph of every page, because the content is literally broken into chunks.

Manetta: Can you say maybe more about this notion of a "polyfill"?

Alex: I think the name comes from polyfilla. The thing you put in the wall, no?

Simon: Oh like when you get a crack in the wall? Polyfill, yes, it's a brand. Yes it's a brand for fixing cracks in the wall.

Alex: So it's exactly that, this idea to fix cracks in the wall.

Simon: Never thought about that.

Alex: Yes the idea is that, correct me if I'm wrong but, so like... you write your code as if you were using natively the functionality, but in the background there is some Javascript or a set of assets, that kind of turn it into a compatible mode for you.

Manetta: And this brought the CSS Regions back?

Alex: Briefly, but then, like I said, there was this speed issue. It was really a mess to layout the magazine we were working on, Médor, with this polyfill. It was really really slow. It was kind of breaking this interactivity we had with the browser.

Doriane: And also, there is an important difference with the polyfill. It tries to replace the way how CSS Regions work, but in doing so it totally changes the way that CSS Regions are working. Because CSS Regions is this kind of illusion, that is rendering content like it was inside the document. And the polyfill is actually breaking up the content and actually putting it into the `<div>`.

Manetta: Did you work with the polyfill for a while, or what happened?

Alex: In my case for a couple of weeks. And then I gave up and we tried to look for other WebKit engines, because actually there were some. I remember using another browser for a while: Epiphany.

Manetta: Which also uses WebKit?

Alex: Yes at least at that time it was using WebKit. And there were some others. But the problem was that the projects were not so active. And sometimes they lack very much on the trunk of the WebKit engine.

Gijs: Yes so there's the difference between the browser and the engine, the browser being the interface and the engine translating the instructions. Just to explain what you said about the trunk and lagging behind.

So what it means to lag behind, is that you work with an old version of the engine. Meanwhile time goes on and new exciting CSS properties emerge, that you cannot use, because the engine is too old, in the sense that it is not updated. So when an engine is lagging behind for a year, you can bump into unexpected surprises, which force you to think why some specific CSS properties are suddenly not working.

Manetta: In the end you forked a browser engine yourself, right?

Alex: Not a browser engine, but... So actually when we did this review of all the browsers using the WebKit engine, at some point we found one, but it was not a browser. It was a wrapper around the WebKit engine, that allowed you to insert a kind of widget into your program, with a web view.

The project we found is called Qt-WebKit. And at some point we got enthusiastic about it and started to make a "web browser" – I'm using quotes because it's a very very minimal one. It is basically a software that has a big web view and a small URL bar. And you click OK and then you can see the page inside the web view. And that is what we called OSPKit, which is part of our html2print workflows.

Manetta: And because OSPKit is based on WebKit, it brought the CSS Regions back?

Alex: Yes. And the developer of Qt-WebKit was still trying to keep the thing updated. And it also was someone who we could reach on IRC and discuss with. I remember once I asked him if there was a specific property available in the browser, and he said no. And 3 minutes later he implemented it for me. So it was a very nice experience, to be so close to the developer and his project.

Manetta: And why was it important to keep working with CSS Regions?

Gijs: So we had developed more and more projects around using CSS Regions, or that were depending on CSS Regions.

Manetta: One of the recurrent projects in which you worked with CSS Regions was Médor, right?

Amélie: Yes so Médor is a Belgian magazine, that is about... I'm not sure how to say it in English. It's journalism and a news magazine, doing deep investigation. There is an issue every three months and it has been laid out with html2print since the beginning.

Manetta: So it was an important project for which you needed OSPKit?

Alex: Yes. I think the first issue was in 2015, so it was really at the time when we were very active about building our toolset. The Médor project both benefited from our research and also was a drive to conduct more research. And because it was ambitious, not in the sense of aesthetics or whatever –it was that as well I hope– but it was ambitious in the sense that the magazine was broadly distributed and reaching a lot of people. So there was a lot of pressure to make sure that we have a correct PDF at the printer in time. Because in journalism the release is a very important milestone that you cannot really miss.

Manetta: Do you want to say more about that question why it was then important to develop OSPKit?

Gijs: If we hadn't done that it wouldn't have been possible to continue working with our workflow. It would have fallen apart and we would have had to rethink completely how we would make the layout for Médor. The layout of Médor is very much based on a grid, using all the boxes and all the space that is available on the page. And without CSS Regions it would not have been possible to produce such layout at that moment. We would have only been able to work with a single flow. You can maybe float elements to the left and right, but that is it. State of the art at that moment were multi-column layouts, and this was often not supported in html2print. Which means that you're left with a very impoverished experience.

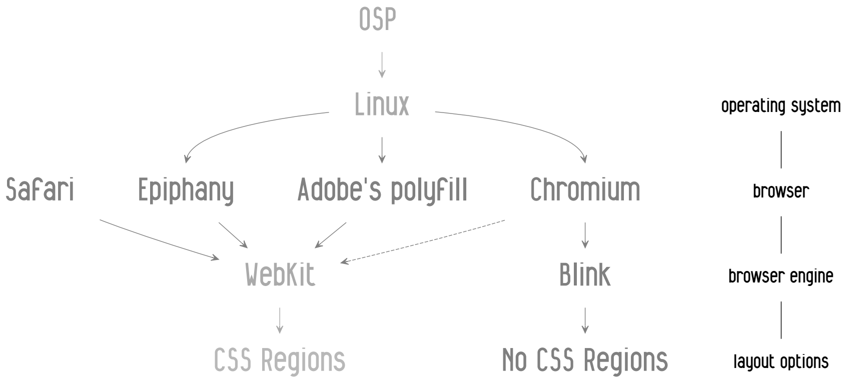
And there's also something about... it being possible. Like you're also maybe clinging on to the possibilities of the moment. In the sense that... I think it's important to mention that there is this promise of open source, that you are able to influence or control your tool. But here it became very clear that a browser engine is such a complex piece of software, and so many people are working on it, and if those people decide to take a different direction, that they don't care about the things that you care about, for whatever reason. This might feel very foreign or might also feel wrong. But it sort of leaves you in the dark. You're there, and the browser caravan is carrying on, following their own path. And you try everything you can to keep on working with it, as long as you can. Also from the hope that, you know... that in WebKit, the CSS Regions remain supported.

Manetta: I'm curious to understand the impact of such workarounds on a design practice like yours. Because in the end OSPKit is a workaround, no? A work around the main direction of the development of the web. A work around the decisions that the makers of browsers make.

What happens when you introduce such workarounds into a design practice? Because it is quite something. Can we unpack that?

Doriane: Yes, maybe. One of the things is that it creates a bit of an alternate reality. Because you're suddenly living in your own browser. The path is split in two. And the current status of web-to-print goes further and new things are happening there. But in the world of this OSPKit browser, things are a bit stuck in time. And okey you have this work around that allows you to use a magic property that you want to keep close to yourself. But then you have to live in your own reality, that is a bit outside of the evolution and the tendency of the rest of the designer practice in web-to-print specifically.

Alex: Yes exactly... Because now



OSPKit is kind of fixed in time, and it's already static since 2016 or something. It's getting very old, especially in this world.

[laughter]

Alex: It was a way to keep a feature feature alive, a very nice feature, or at least a work around that allowed us to stay with our practice. But at the same time it's also, like you said, it is cutting us from new practices that could arise, with new web CSS properties and developments of the web. So yes, it's a bit, I don't know how to say it, but it's doing good and bad at the same time.

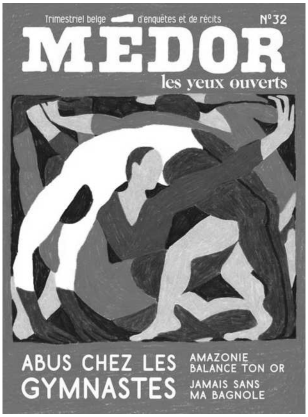
Amélie: Just a few hours before the interview we were chatting and Gijs used the word *technological archeology*, and I think it fits to the way I feel as I'm coming back on Médor and I didn't especially follow the development of html2print. Yes that's it. I'm using that tool, that's using old technologies, and we cannot use more recent CSS properties. And so yes, we have to work in other ways and find other ways of doing.

Sometimes I'm trying to do something, and then I realise, oh I cannot use the recent CSS, so let's try to find another way to do it otherwise. It's another mindset.

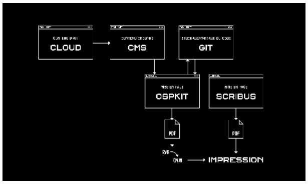
Doriane: Yeah and it's a weird feeling. Like when you're used to moments when you think, oh I don't know how to do this thing, then you're looking at the docs online, and then you're doing it. And of course it's working, because you copy paste the example from the doc. But then you cannot just look at the doc, you need to test everything and if something is not working you're not sure what exactly is working and what not.

I remember that especially when working with Javascript, realising that yes, we're stuck with using a version of Javascript of 2016, which has evolved a lot since. And it's also different to work with HTML and CSS from 2016.

For example, when you want to make a border around a font, and the border does not show, you know that this CSS property was not supported in 2016. But if you're writing Javascript it becomes super hard to debug, because you have no idea which line is supported and which one not.



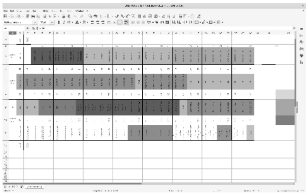
SWITCHING TO THE OSP STUDIO AGAIN, ON A FRIDAY MORNING IN JUNE 2023. Alex joined me almost every Friday, to dive into OSPKit and CSS Regions together. I was curious to talk a bit more about the use of the CSS Regions in the making of all those issues of the Médor magazine, since 2015, which is 8 years already!



Graph of the Médor workflow, which can be summarized as: Nextcloud → cms (Wagtail) → OSPKit ↔ git ↔ OSPKit → PDF + Scribus → PDF.

Are there other tools used to organise the collaboration between the designers, the editors and the journalists?

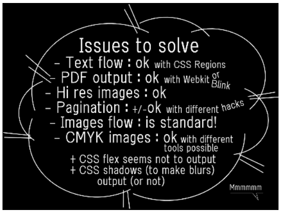
We always needed to make a flatplan of the magazine, to see if there was no conflict between let's say two articles. Or just to see how we want to spread the content. To, for example, not have too much depressing investigations around corruption at the same place in the magazine. The journalists came up with a simple solution that we still use actually, which is a simple spreadsheet with 128 cells to dispatch the pages of the magazine.



The spreadsheet that is used for the flatplan.

Is such a flatplan an important tool for the overall workflow? Is it important for you for example to know on which page an article starts?

It's important in the sense that we specify the pagination by hand. And we have to know if an article starts on the left of right page.



Issues to solve
- Text flow : ok with CSS Regions
- PDF output : ok with Webkit or Blink
- Hi res images : ok
- Pagination : +/-ok with different hacks
- Images flow : is standard!
- CMYK images : ok with different tools possible
+ CSS flex seems not to output
+ CSS shadows (to make blurs) output (or not)
Mmmmmm

SWITCHING TO THE OSP STUDIO, MAY 2023,

WHERE I AM IN THE MIDDLE OF MY 3 MONTHS RESIDENCY, EXPLORING THE IMPACT OF CSS REGIONS ON THE PRACTICE OF OSP.

I would like to publish a post on the OSP blog, to dive further into the timelines of the CSS Regions. But hmm, how can I introduce readers to this story? Where to start?

This is what I eventually wrote and published on the OSP blog here: http://blog.osp.kitchen/residency/from-webkit-to-ospkit.html.

In 2013, in the middle of W3C discussion threads around

And potentially, to have an overview to see how the articles are spread, in terms of rhythm and stuff.

And also, it can be nice sometimes to know where the center folds are, for example when you have images across two pages.

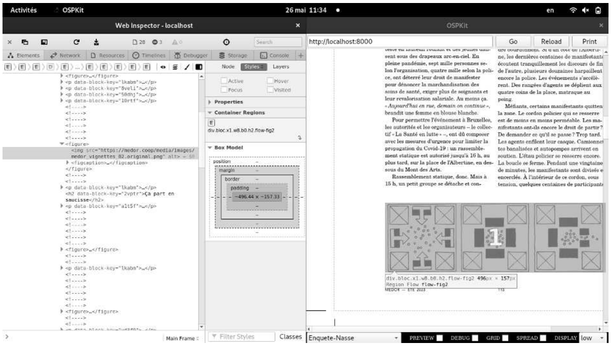
How do you connect the articles with OSPKit?

There is an API. In the former website it used to be a simple HTML output. We were just taking the article from the CMS and it almost had not structure, just a few classes. And now it's a json API.

Do you have an example in which we can see the CSS Regions at work?

For this article [on the screenshot below] there was a mix of images that should appear inside the body text, in the columns, and others were placed on specific positions.

This image for example is placed at a specific position, at the bottom of the page.



```
<figure>…</figure>
<p data-block-key="lkabm">…</p>
<p data-block-key="8vell">…</p>
<p data-block-key="58dhj">…</p>
<p data-block-key="10rtf">…</p>
<!---->
<!---->
<!---->
<!---->
<!---->
<!---->
<figure>
  <img src="https://medor.coop/media/images/medor_vignettes_82.original.png" alt> = $0
  <figcaption>…</figcaption>
</figure>
<!---->
<!---->
<p data-block-key="lkabm">…</p>
<h2 data-block-key="2vptr">Ça part en saucisse</h2>
<p data-block-key="a1t5f">…</p>
<!---->
<!---->
<!---->
<!---->
<!---->
<!---->
<figure>…</figure>
<!---->
<!---->
<p data-block-key="lkabm">…</p>
<!---->
<!---->
<!---->
<!---->
<!---->
<!---->
<figure>…</figure>
<!---->
<!---->
```

Container Regions: div.bloc.x1.w8.b0.h2.flow-fig2

Box Model: −496.44 × −157.33

http://localhost:8000

sent sous des drapeaux arc-en-ciel. En pleine pandémie, sept mille personnes selon l'organisation, quatre mille selon la police, ont déterré leur droit de manifester pour dénoncer la marchandisation des soins de santé, exiger plus de soignants et leur revalorisation salariale. Au moins ça. « *Aujourd'hui en rue, demain on continue* », brandit une femme en blouse blanche.

Pour permettre l'événement à Bruxelles, les autorités et les organisateurs – le collectif « La Santé en lutte » –, ont dû composer avec les mesures d'urgence pour limiter la propagation du Covid-19 : un rassemblement statique est autorisé jusqu'à 16 h, au plus tard, sur la place de l'Albertine, en dessous du Mont des Arts.

Rassemblement statique, donc. Mais à 15 h, un petit groupe se détache et con-

ne, les dernières centaines de manifestants écoutent tranquillement les discours de fin de l'autre, plusieurs douzaines harpaillent encore la police. Les événements s'accélèrent. Des rangées d'agents se déplient aux quatre coins de la place, matraque au poing.

Méfiants, certains manifestants quitten la zone. Le cordon policier qui se resserre est de moins en moins perméable. Les manifestants ont-ils encore le droit de partir ? De demander ce qu'il se passe ? Trop tard. Les agents enfilent leur casque. Camionnet tes banalisées et autopompes arrivent en soutien. L'étau policier se resserre encore. La boucle se ferme. Pendant une vingtaine de minutes, les manifestants sont divisés e encerclés. À l'intérieur de ce cordon, sous tension, quelques centaines de participant

div.bloc.x1.w8.b0.h2.flow-fig2 496px × 157px
Region Flow flow-fig2

MEDOR — ÉTÉ 2023 115

Enquete-Nasse

DISPLAY low

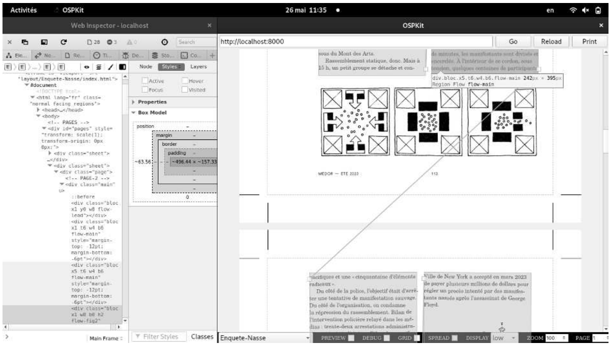
Image positioned at a predefined place: at the bottom of the page.

And if you go to "computed" you have a very feature to see from which region it is flowing. So if I click there, it will bring me to the div element that contains the flow. Not from where it originates but to where it flows.

Paged Media features of the web, browsers engine shifts and partially implemented CSS standards, OSP presented at the Libre Graphics Meeting in Madrid, to share insights and excitement for using HTML and CSS to make books and publications.

While figuring out what impact it would have on their practice to make printed matter with web technologies, OSP listed their *issues to solve*. At the top of this list we find the issue of flowing text on a page. Which is marked as resolved with an *"ok"*, thanks to the presence of a specific CSS property: CSS Regions.

The *Mmmmmm* at the right bottom of the slide might already indicate a gut feeling and awareness of the always-changing dynamics of the web. In the same year, in 2013, Chromium announced



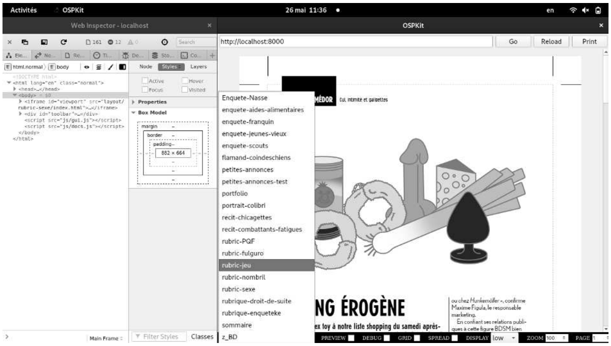
```
'layout/Enquete-Nasse/index.html">
  #document
    <!DOCTYPE html>
    <html lang="fr" class=
    "normal facing regions">
      <head>…</head>
      <body>
        <!-- PAGES -->
        <div id="pages" style=
        "transform: scale(1);
        transform-origin: 0px
        0px;">
          <div class="sheet">
          …</div>
          <div class="sheet">
            <div class="page">
              <!-- PAGE-2 -->
              <div class="main"
              u>
                ::before
                <div class="bloc
                x1 y0 w8 flow-
                lead"></div>
                <div class="bloc
                x1 t6 w4 b6
                flow-main"
                style="margin-
                top: -12pt;
                margin-bottom:
                -6pt"></div>
                <div class="bloc
                x5 t6 w4 b6
                flow-main"
                style="margin-
                top: -12pt;
                margin-bottom:
                -6pt"></div>
                <div class="bloc
                x1 w8 b8 h2
                flow-fig2"
```

```
div.bloc.x5.t6.w4.b6.flow-main 242px × 395px
Region Flow flow-main
```

Text flowing into different <div> elements.

Ah nice you see the text flowing from one page to the other here, nice!

Maybe in another article.



```
<!DOCTYPE html>
<html lang="en" class="normal">
  <head>…</head>
  <body> = $0
    <iframe id="viewport" src="layout/
    rubric-sexe/index.html">…</iframe>
    <div id="toolbar">…</div>
    <script src="js/gui.js"></script>
    <script src="js/docs.js"></script>
  </body>
</html>
```

Screenshot of OSPKit, Alex switches to another article.

Is it more difficult for you to work with html2print when there is a strict seperation between content and form? I mean, if you want to change the content, you need to make an edit in the CMS right?

Yes but since we have the CSS Regions we can still move things around.

Let's say we have a side note in an article, this side note does not have to be encoded at this specific place in the CMS.

Do CSS Regions bring back, in a way, the ability to work with both content and form at the same time?

that they will switch browser engine, from WebKit to Blink, and that they will drop the support of CSS Regions. Since then, it has become increasingly hard to still use CSS Regions on Linux machines. But not impossible.

The story around OSP's work with CSS Regions introduces a particular example of a dependency relation that is entangled with a complex configuration of software timelines, web standards, layout traditions, commissioned work and excitement to explore book making with HTML and CSS.

Why did OSP choose to stay with a never fully implemented CSS standard? In what degree are the CSS Regions important for OSP? Which workarounds are needed to keep working with CSS Regions today?

Yes it allows to move things around and change the structure that is hardcoded and very linear in HTML. Sometimes it is more practical that a text flows in order, but this order does not be the final order in the layout.

What is the difference between CSS Regions and CSS grid? Both of them can be used to flow content into a template, right?

CSS Grid positions already fragmented content. When something overflows, it's hidden, it does not flow into another div. It does not fragment content, it only positions it.

What are the limits of CSS Regions? When do you switch to another tool, like Scribus?

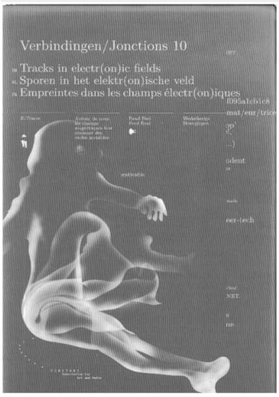
Usually when we switch it is about color. What happens sometimes is that we have illustrations with spot colors, or not even spot colors, but with precise CMYK values.

It's interesting that the workflow enables you to work with different tools. Was it a conscious choice to do it in this way?

At first there was this disappointment with… or let's turn it positively… there was this excitement with using HTML.

But i think the mix of software also came back because we could see that each software has certain weak points and strengths, or features. I think for me something that is very important in OSP is that we embrace the diversity of tools. We have never been so excited about replacing X with Y. Replacing illustrator with Inkscape for example. There was the excitement to use different tools for different kind of things. Graphviz is for example a good example.

And for layout, i still like the idea that you can use different tools.


Verbindingen/Jonctions 10
Tracks in electr(on)ic fields
Sporen in het elektr(on)ische veld
Empreintes dans les champs électr(on)iques

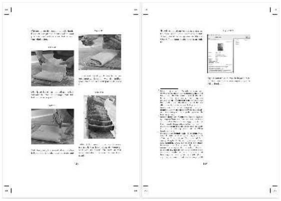
SWITCHING TO THE SPECULOOS STUDIO, WHERE PIERRE TAKES *TRACKS IN ELECTR(ON)IC FIELDS* IN HIS HANDS AND STARTS BROWSING...

In the case of *Tracks in electr(on)ic fields*, the book is made with ConTeXt, which works through a process of compilation, so it's the same with html2print. And at some point your file is not compiling, and all the stuff, you know… For me, and so for Femke, it was the first experience being so frustrated. To go to the author of ConTeXt, Hans Hagen, in the Netherlands, to ask him to help us… Overall, clearly for me it was super traumatic as an experience. Because also at some point, my own limitiations to work with abstractions of the language, computer language I mean, was really blatent and in many occassions I have been

obliged to let Femke alone. It was also very painful to see that. And you don’t know what to do.

m: Because it was you and Femke working on the book?

p: Yes yes. So the classic but frustrating for both sides, when the developer part is split. At some point Femke asked some help from Michael, but Michael was only helping on the principles of extreme computing and like that, but not in the core problems that we had. So it has been solved mainly by Femke alone.



m: Do you remember what the problems were? There were multiple problems right?

p: That voodoo stuff with TeX. What is bizarre with TeX is that it needs multiple loops to finalize a compilation. So it bizarrely processes a file and reprocesses it with stuff that it has gained in the first iteration. And some stuff goes voodoo. Some stuff are breaking and you see randomly there is stuff that go outside the boxes, that is

SWITCHING TO A SMALL GARDEN OF THE SPECULOOS STUDIO IN BRUSSELS, WHERE I'M SITTING ON A WOODEN BENCH WITH PIERRE HUYGENBAERT.

It's July 2023. A big table is standing in front of us. Pierre just took a moment to search in the Speculoos book shelves for some books made by OSP throughout the years, which now has created a nice messy pile of a lot of different kinds of publications.

p: It was after several attempts by OSP and Femke seperately, but mainly by OSP, to use Scribus for long texts, for books. And it was painful. After the really beginning of OSP in 2006, 2007, 2008, many works were made in Scribus. We were always in and out with Scribus, but it was really painful to work with long documents, because at the time it was a very bizarre behavior of Scribus, that made files grow exponentialy in size with the page count. I do not remember at what moment precisely, but Andreas Vox, the Scribus dark intelligence, has admitted it at some point.

m: The more pages there are in your Scribus document, the bigger the size of the file?

p: Yes, very bizarrely. Because there is a kind of a loop inside that process. And so for that kind of stuff you are really obliged todiv ide your Scribus document into multiple. So with Femke we decided to go elsewhere, which was TeX. I don’t remember how much the text in *Tracks in electr(on)ic fields* describes how much it has been super painful. It was even more.

But let’s say, with Scribus it was more and more slow, and in the end it was a half gigabyte file to work with, but you are never stuck to the point that it’s not giving output. You feel it’s stupid and the slowness is painful, but you are not really stuck. You are working with full breaks on, but not stuck.

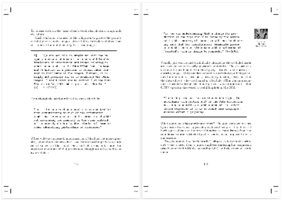
not really a big problem, but also all the indexation system that has gone really another way. In the end it was beautiful, but it was not the way we wanted to do it.

p: [browsing the book]

p: And also it was really frustrating to see how we hang on the fact that it was not possible to change the font. Each time we changed the font it just exploded.

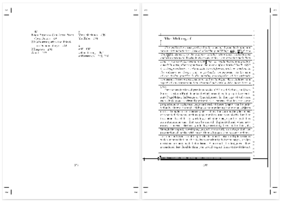
m: It didn't compile?

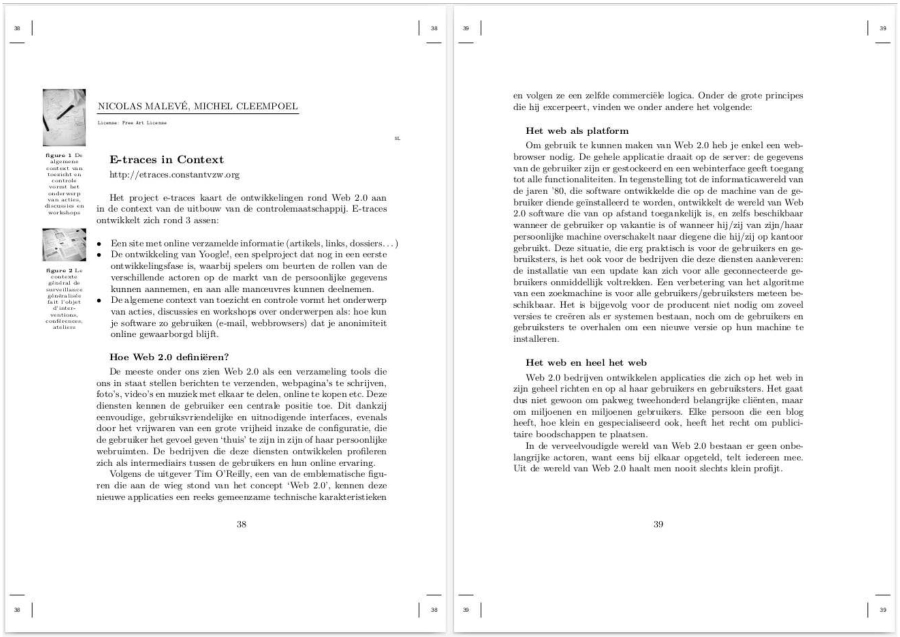
p: Yes. Or in a crazy way, or… So many dimensions of unpredictability. I think it has provoked my reluctance to go to other kinds of compilation based ways of working.



m: But you came close to them when making the Balsamine programme booklet for 2013-2014, right?

p: Yes, this was the first one that was made with HTML and CSS, but also many other things: Scribus, Python, Graphviz, Ghostscript.





38 | 38

NICOLAS MALEVÉ, MICHEL CLEEMPOEL

License: Free Art License

NL

figure 1 De algemene context van toezicht en controle vormt het onderwerp van acties, discussies en workshops

figure 2 Le contexte général de surveillance généralisée fait l'objet d'interventions, conférences, ateliers

E-traces in Context

http://etraces.constantvzw.org

Het project e-traces kaart de ontwikkelingen rond Web 2.0 aan in de context van de uitbouw van de controlemaatschappij. E-traces ontwikkelt zich rond 3 assen:

- Een site met online verzamelde informatie (artikels, links, dossiers. . .)
- De ontwikkeling van Yoogle!, een spelproject dat nog in een eerste ontwikkelingsfase is, waarbij spelers om beurten de rollen van de verschillende actoren op de markt van de persoonlijke gegevens kunnen aannemen, en aan alle manœuvres kunnen deelnemen.
- De algemene context van toezicht en controle vormt het onderwerp van acties, discussies en workshops over onderwerpen als: hoe kun je software zo gebruiken (e-mail, webbrowsers) dat je anonimiteit online gewaarborgd blijft.

Hoe Web 2.0 definiëren?

De meeste onder ons zien Web 2.0 als een verzameling tools die ons in staat stellen berichten te verzenden, webpagina's te schrijven, foto's, video's en muziek met elkaar te delen, online te kopen etc. Deze diensten kennen de gebruiker een centrale positie toe. Dit dankzij eenvoudige, gebruiksvriendelijke en uitnodigende interfaces, evenals door het vrijwaren van een grote vrijheid inzake de configuratie, die de gebruiker het gevoel geven 'thuis' te zijn in zijn of haar persoonlijke webruimten. De bedrijven die deze diensten ontwikkelen profileren zich als intermediairs tussen de gebruikers en hun online ervaring.

Volgens de uitgever Tim O'Reilly, een van de emblematische figuren die aan de wieg stond van het concept 'Web 2.0', kennen deze nieuwe applicaties een reeks gemeenzame technische karakteristieken

38

38 | 38

39 | 39

en volgen ze een zelfde commerciële logica. Onder de grote principes die hij excerpeert, vinden we onder andere het volgende:

Het web als platform

Om gebruik te kunnen maken van Web 2.0 heb je enkel een webbrowser nodig. De gehele applicatie draait op de server: de gegevens van de gebruiker zijn er gestockeerd en een webinterface geeft toegang tot alle functionaliteiten. In tegenstelling tot de informaticawereld van de jaren '80, die software ontwikkelde die op de machine van de gebruiker diende geïnstalleerd te worden, ontwikkelt de wereld van Web 2.0 software die van op afstand toegankelijk is, en zelfs beschikbaar wanneer de gebruiker op vakantie is of wanneer hij/zij van zijn/haar persoonlijke machine overschakelt naar diegene die hij/zij op kantoor gebruikt. Deze situatie, die erg praktisch is voor de gebruikers en gebruiksters, is het ook voor de bedrijven die deze diensten aanleveren: de installatie van een update kan zich voor alle geconnecteerde gebruikers onmiddellijk voltrekken. Een verbetering van het algoritme van een zoekmachine is voor alle gebruikers/gebruiksters meteen beschikbaar. Het is bijgevolg voor de producent niet nodig om zoveel versies te creëren als er systemen bestaan, noch om de gebruikers en gebruiksters te overhalen om een nieuwe versie op hun machine te installeren.

Het web en heel het web

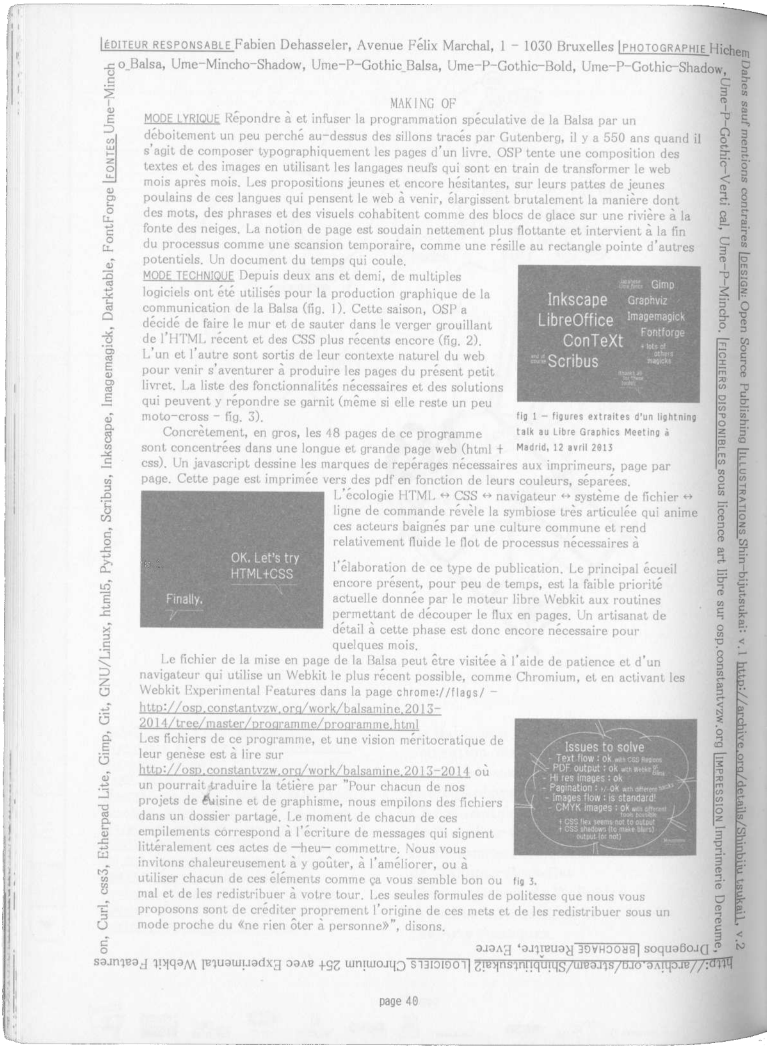
Web 2.0 bedrijven ontwikkelen applicaties die zich op het web in zijn geheel richten en op al haar gebruikers en gebruiksters. Het gaat dus niet gewoon om pakweg tweehonderd belangrijke cliënten, maar om miljoenen en miljoenen gebruikers. Elke persoon die een blog heeft, hoe klein en gespecialiseerd ook, heeft het recht om publicitaire boodschappen te plaatsen.

In de verveelvoudigde wereld van Web 2.0 bestaan er geen onbelangrijke actoren, want eens bij elkaar opgeteld, telt iedereen mee. Uit de wereld van Web 2.0 haalt men nooit slechts klein profijt.

39

39 | 39

SWITCHING TO THE *MAKING OF*, WRITTEN BY OSP AND PUBLISHED IN THE BACK OF THE BALSAMINE PROGRAMME BOOKLET (2013-2014).



ÉDITEUR RESPONSABLE Fabien Dehasseler, Avenue Félix Marchal, 1 – 1030 Bruxelles PHOTOGRAPHIE Hichem Dahes sauf mentions contraires DESIGN: Open Source Publishing ILLUSTRATIONS Shin-bijutsukai: v.1 http://archive.org/details/Shinbiju tsukai1, v.2 http://archive.org/stream/Shinbijutsukai2 LOGICIELS Chromium 25+ avec Experimental Webkit Features on, Curl, css3, Etherpad Lite, Gimp, Git, GNU/Linux, html5, Python, Scribus, Inkscape, Imagemagick, Darktable, FontForge FONTES Ume-Mincho_Balsa, Ume-Mincho-Shadow, Ume-P-Gothic_Balsa, Ume-P-Gothic-Bold, Ume-P-Gothic-Shadow, Ume-P-Gothic-Verti cal, Ume-P-Mincho. FICHIERS DISPONIBLES sous licence art libre sur osp.constantvzw.org IMPRESSION Imprimerie Dereume, Drogenbos BROCHAGE Renaître, Evere

MAKING OF

MODE LYRIQUE Répondre à et infuser la programmation spéculative de la Balsa par un déboitement un peu perché au-dessus des sillons tracés par Gutenberg, il y a 550 ans quand il s'agit de composer typographiquement les pages d'un livre. OSP tente une composition des textes et des images en utilisant les langages neufs qui sont en train de transformer le web mois après mois. Les propositions jeunes et encore hésitantes, sur leurs pattes de jeunes poulains de ces langues qui pensent le web à venir, élargissent brutalement la manière dont des mots, des phrases et des visuels cohabitent comme des blocs de glace sur une rivière à la fonte des neiges. La notion de page est soudain nettement plus flottante et intervient à la fin du processus comme une scansion temporaire, comme une résille au rectangle pointe d'autres potentiels. Un document du temps qui coule.

MODE TECHNIQUE Depuis deux ans et demi, de multiples logiciels ont été utilisés pour la production graphique de la communication de la Balsa (fig. 1). Cette saison, OSP a décidé de faire le mur et de sauter dans le verger grouillant de l'HTML récent et des CSS plus récents encore (fig. 2). L'un et l'autre sont sortis de leur contexte naturel du web pour venir s'aventurer à produire les pages du présent petit livret. La liste des fonctionnalités nécessaires et des solutions qui peuvent y répondre se garnit (même si elle reste un peu moto-cross – fig. 3).

Inkscape
LibreOffice
ConTeXt
and of course Scribus
Gimp
Graphviz
Imagemagick
Fontforge
+ lots of others magicks

fig 1 – figures extraites d'un lightning talk au Libre Graphics Meeting à Madrid, 12 avril 2013

Concrètement, en gros, les 48 pages de ce programme sont concentrées dans une longue et grande page web (html + css). Un javascript dessine les marques de repérages nécessaires aux imprimeurs, page par page. Cette page est imprimée vers des pdf en fonction de leurs couleurs, séparées.

OK. Let's try HTML+CSS
Finally.

L'écologie HTML ↔ CSS ↔ navigateur ↔ système de fichier ↔ ligne de commande révèle la symbiose très articulée qui anime ces acteurs baignés par une culture commune et rend relativement fluide le flot de processus nécessaires à l'élaboration de ce type de publication. Le principal écueil encore présent, pour peu de temps, est la faible priorité actuelle donnée par le moteur libre Webkit aux routines permettant de découper le flux en pages. Un artisanat de détail à cette phase est donc encore nécessaire pour quelques mois.

Le fichier de la mise en page de la Balsa peut être visitée à l'aide de patience et d'un navigateur qui utilise un Webkit le plus récent possible, comme Chromium, et en activant les Webkit Experimental Features dans la page chrome://flags/ – http://osp.constantvzw.org/work/balsamine.2013-2014/tree/master/programme/programme.html

Les fichiers de ce programme, et une vision méritocratique de leur genèse est à lire sur http://osp.constantvzw.org/work/balsamine.2013-2014 où un pourrait traduire la têtière par "Pour chacun de nos projets de cuisine et de graphisme, nous empilons des fichiers dans un dossier partagé. Le moment de chacun de ces empilements correspond à l'écriture de messages qui signent littéralement ces actes de —heu— commettre. Nous vous invitons chaleureusement à y goûter, à l'améliorer, ou à utiliser chacun de ces éléments comme ça vous semble bon ou mal et de les redistribuer à votre tour. Les seules formules de politesse que nous vous proposons sont de créditer proprement l'origine de ces mets et de les redistribuer sous un mode proche du «ne rien ôter à personne»", disons.

Issues to solve
- Text flow : ok with CSS Regions
- PDF output : ok with Webkit or Blink
- Hi res images : ok
- Pagination : +/- ok with different tricks
- Images flow : is standard!
- CMYK images : ok with different tools possible
+ CSS flex seems not to output
+ CSS shadows (to make blurs) output (or not)

fig 3.

page 40

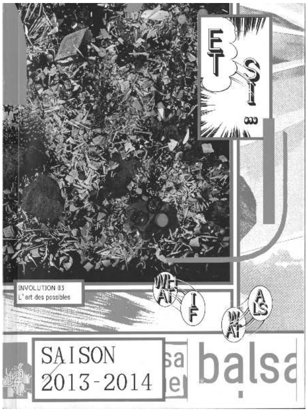
On the next page you find an English translation that i worked on in closeness of a machine translator, as my French is not so advanced, specially not to catch the poetics of this text.

Translated in August 2023.

Responding to and infusing the Balsa's speculative programming with a dislocation slightly perched above the furrows traced by Gutenberg 550 years ago when it came to typographically composing the pages of a book. OSP attempts to compose texts and images using the new languages that are transforming the web month after month. The young and still hesitant proposals, on their feet as the young foals of these languages that are thinking of the web to come, brutally broaden the way in which words, sentences and images live together like blocks of ice on a river as the snow melts. The notion of the page is suddenly much more buoyant, and at the end of the process intervenes like a temporary scansion, like a mesh in the rectangle pointing to other potentials. A document of the passage of time.

Over the last two and a half years, a wide range of software has been used to produce the graphics for La Balsa's communications (fig. 1). This season, OSP has decided to take the plunge and jump into the teeming orchard of recent HTML and even more recent CSS (fig. 2). Both have been taken out of their natural web context to venture into the production of the pages in this little booklet. The list of functionalities required and the solutions that can meet them is growing (even if it remains a little motocross - fig. 3). Basically, the 48 pages of this programme booklet are concentrated in a long, large web page (html + css). A javascript script draws the registration marks needed by printers, page by page. This page is printed to PDF's and separated according to their colours. The HTML <- - > CSS <- - > browser <- - > file system <- - > command line ecology reveals the highly articulated symbiosis that animates these players bathed in a common culture and makes the flow of processes required to produce this type of publication relatively fluid. The main stumbling block that remains, for the foreseeable future, is the low priority currently given by the Webkit open-source engine to routines enabling the stream to be broken down into pages. Some fine-tuning at this stage is therefore still necessary for a few months.

The Balsa layout file can be visited with patience and a browser that uses the most recent Webkit possible, such as Chromium, and by activating the Webkit Experimental Features at the chrome://flags page.



SWITCHING TO THE GARDEN OF SPECULOOS, WHERE PIERRE MEANWHILE BROWSES THROUGH THE BALSAMINE BOOKLET AND STOPS AT THE *MAKING OF* TEXT...

p: Wow, the *Making of* text is really high level abstract, even in French. It is probably partly linked to the Gutenberg traces that we encountered in the making of the *Tracks in electr(on)ic fields* publication. I'm not sure if Knuth, the person behind TeX, refers to Gutenberg directly, but if I remember it well, Hans Hagen was referring to that. And you feel it everywhere. So the understanding of Gutenberg's legacy is probably infused with that.

m: Are fixed margins on all the pages of a book one example of a Gutenberg legacy?

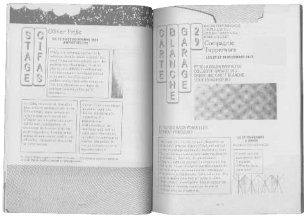
p: Yes, the margins were quite difficult to change in fact. We were fighting with that. Then the font… Many things were quite resisting any change in fact. Like it was hard coded.

m: And running headers?

p: This is not Gutenberg in fact, it's an idea of how a document should look.

m: In a way Gutenberg is also an approach, no? It's clear that legibility is a really important aspect within this way of thinking. Maybe we can try to formulate differences? What was for example different between the making of *Tracks in electr(on)ic fields* using ConTeXt and the Balsa booklet using HTML and CSS?

p: I think, for me when I have seen the Balsa booklet taking shape, which was not too far from here actually, it was at Ludi's place. What was really exciting, was to really code it in HTML and CSS. To see the flow and to say, oke, break that in this and this and this. That kind of stuff was not possible to do in other software. And immediately there were also frustrations when it did not work, when it breaks things, but it still was like a [wow]. It is probably the excitement of making things in another way.



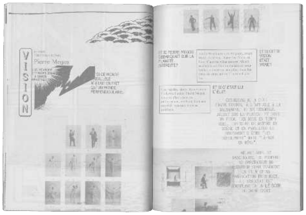
p: What was super exciting in this stuff, was the fact that you feel the flow even if that flow is articulated or broken in a very specific way. And also to know that it was a flow. Or it's a flow that is guided through a quite detailed secrecy. At that moment it was feeling like, okay, we are there, we found something super exciting, yet it's only the

beginning. But now some of our problems are behind us.

m: You mean some of the problems with Scribus and ConTeXt?

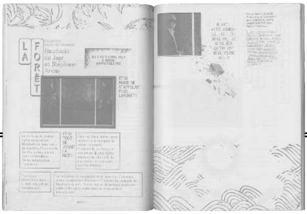
p: Yes. It was then possible to have at the same time a flow, with that practicality of efficiency and being able to lay out it in a

different way for the web, etcetera. And at the same time it's something that is not looking like a flow, with all the clumsiness we want or need, and with the blocks, without the painful approach of working with a canvas to pick every part… Which is like layouting stuff in Photoshop…



p: Working with HTML for print for Balsa 2013-2014 was super super exciting, I remember. And, yes after, we could see that, for me, and I might be wrong… I never felt again that kind of excitement…

All things afterwards were more disappointing, because the clash with the CSS standards, the fact that the browsers at that moment were so full of promises, and not the kind of dissection we have seen afterwards with Google optimizing and stuff, removing possibilities that seem crazy.



SWITCHING TO A BLOG POST PUBLISHED IN JANUARY 2014, WHERE HÅKON WIUM LIE, ONE OF THE PEOPLE WHO WERE INVOLVED IN THE INITIAL WRITING OF CSS, CRITICIZES THE IDEA OF THE CSS REGIONS.

What Pierre refers to is the moment that Google removed support for CSS Regions from their Chrome/Chromium browser, and thus removed the possibility to work with multiple flows and to position these flows on a page manually. It turned the situation back to being restricted to work with a single flow. The single flow became the default again, which means that your material can only be laid out in a certain order, from top to bottom. You forget how things are flowing into a page. You forget about the underlying fragmentation, the blocks and the flow logics of HTML, …

THE A LIST APART BLOG PRESENTS:

CSS Regions Considered Harmful

by **Håkon Wium Lie** · January 22, 2014

Published in Uncategorized

Håkon Wium Lie is the father of CSS, the CTO of Opera, and a pioneer advocate for web standards. His last article in this magazine led directly to real fonts on the web. When Håkon speaks, whether we always agree or not, we listen. In today's post, Håkon shares his opinion on CSS Regions.

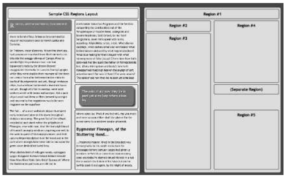
(...)

Computer scientists have a peculiar way of expressing fear and doubt. They publish essays with “considered harmful” in the title. This particular design pattern was started by Edsger Dijkstra when he published “Go To Statements Considered Harmful” in 1968. The development of formatting objects led me to use the same device; “Formatting Objects Considered Harmful” argued that formatting objects were `font` tags in disguise and that their use on the web must be avoided to preserve web semantics.

It seems that proposals for presentational elements return every so often. The most recent incarnation is CSS Regions. One should not write “considered harmful” articles lightly, but presentational elements is not the only problem with CSS Regions. For those who believe in meaningful HTML tags, responsive web design, and compact CSS code, the introduction of CSS Regions is not good news.

Problem #1: regions use dummy divs

Some articles on CSS Regions have already looked at the source code. An article published by WebPlatform.org describes how to achieve a commonly used two-column design:



The formatted document is on the left, and the corresponding regions are shown on the right. The HTML code that generates this layout must be studied in order to understand CSS Regions. Here's a snippet:

```
<section class="page">
  <div id="title"><h1>Region#1</h1></div>
  <div id="intro"><h1>Region#2</h1></div>
  <div id="col1"><h1>Region#3</h1></div>
  <div id="col2a"><h1>Region#4</h1></div>
  <div id="pull"><h1>(SeparateRegion)</h1
  <div id="col2b"><h1>Region#5</h1></div>
</section>
```

The elements above represent regions, which are containers where text can flow from one to the other. Here is some of the corresponding CSS declarations for the `#intro` element:

```
#intro {
       width: 45%;
       position: absolute;
       top: 5em;
       height: 3em;
       -webkit-flow-from: main;
       -ms-flow-from: main;
       flow-from: main;
}
```

The CSS code above says, roughly: turn the `#intro` element into an absolutely positioned element with a given size and position, then discard the content of the element and replace it with content from the flow called “main”. Thus, the h1 element inside `#intro` isn't a headline at all—the `div` element is a presentational container and the h1 element is discarded.

The proponents of CSS Regions might argue that, "Yes, the divs are there for presentational purposes, but only elements can be scripted on the web and we must therefore use elements." This underlines an important point: it's not regions per se that are harmful to web semantics, it's the fact that they are encoded as presentational HTML elements. If we want regions on the web, we should find a way to write them in CSS and not in HTML. If CSS Regions are accepted in 2014, we will be stuck with absolutely positioned dummy divs for the foreseeable future.

Problem #2: regions are not responsive

Responsive design is a hallmark of good web design. We want our sites to be scalable across a wide range of devices; from small mobile phones, to smarter phones, to big screens.

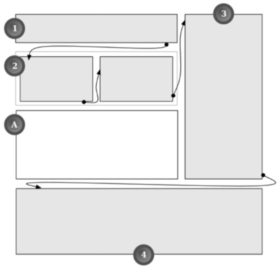
(...)

Ideally, you want the number of columns to be dynamic so that a narrow screen has one column, a medium screen has two columns, and an ultra-wide screen has three, or maybe four columns. CSS Regions will not give you this.

(...)

Problem #3: confusing text flow

Specifications often start out with a motivational example to show how powerful the proposed functionality is and how easy it can be achieved. The first example of the CSS Regions specification is shown below:



The text flow moves from region 1, to 2, 3, and 4, following the arrows. Notice how the eyes of the reader will have to traverse sideways, in the opposite of the reading direction, from 3 to 4. These kinds of traversals are not common in newspaper design, and I will argue that they are confusing to readers and should be avoided. CSS Multi-column Layout cannot make text flow this way, and I consider that to be a feature.

(...)

Amongst the problems discussed in this article, this is probably the one I'm least worried about; if enough users are confused, the design will change. But it seems wasteful to invest years of efforts to implement CSS Regions if most of the compelling use cases can be achieved through an existing mechanism.

Problem #4: verbosity

(...)

How many lines would it take to encode the more intuitive design (shown in the figure above) using CSS Multi-column Layout? Three, it turns out:

```
article { columns: 20em }
h1 { column-span: 2 }
img { column-span: 2; float: bottom }
```

For this example, using CSS Regions is a magnitude more complex than using CSS Multi-column Layout. If one were to support one layout on small screens, and more columns on wider screens, the code size for using CSS Regions would grow quickly.

(...)

Problem #5: code reuse

(...)

Stylesheets written this way will not be reusable, each new document will have its own.

Harmful?

CSS Regions were proposed by Adobe in 2011 and the company is still its main proponent. It's laudable that Adobe takes the web seriously and that it brings proposals to W3C—certainly much better than pushing a proprietary technology like Flash. Its motivation is to sell authoring tools that generate CSS code. That's good, too—the web needs good authoring tools and Adobe can make them. But CSS Regions, as currently proposed, will not improve the web. Rather, it brings presentational tags, verbose code, and per-document stylesheets.

SWITCHING TO A XMPP CHAT WITH MICHAEL, WHERE WE STARTED TO TALK ABOUT THE "CONSIDERED HARMFUL" PHRASE. He responded very enthusiastically when i told him that i was working on this. I did not fully understand what made him *so* enthusiastic at first... So i curiously took a dive into the "considered harmful" timelines a bit.

michael@lurk.org: https://homepages.cwi.nl/~storm/teaching/reader/Dijkstra68.pdf

michael@lurk.org: This is the infamous Dijkstra article

mb@vvvvvvaria.org: Ah yes, Gijs pointed me to it too, but thanks for resending it, i lost the link

michael@lurk.org: And what I was looking at is Simon yuills articles in software studies a lexicon but its less related to Dijkstra than I thought

michael@lurk.org: But I know that Simon Yuill is really interested in Dijkstra and once presented hand written papers from Dijkstra at a work session, possibly promiscuous pipelines, as he has there, but I don't have linka

michael@lurk.org: Linka

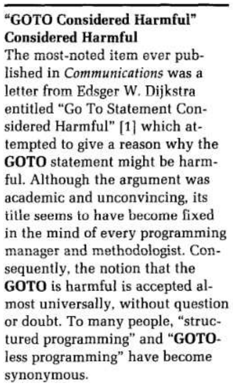
michael@lurk.org: Links

michael@lurk.org: Ugh

mb@vvvvvvaria.org: I just found this response to Dijkstra's GOTO considered harmful published in 1987: https://web.archive.org/web/20090320002214/http://www.ecn.purdue.edu/ParaMount/papers/rubin87goto.pdf

mb@vvvvvvaria.org: The *"harder to create, test and modify"* is super interesting! It makes me think again about our previous conversations around the pedagogical space of bash and the thinking of Papert.

SWITCHING TO A REVIEW WRITTEN BY FRANK RUBIN PUBLISHED IN 1987, IN WHICH HE IS CRITICAL ABOUT THE "CONSIDERED HARMFUL" PHRASE USED BY EDSGER DIJKSTRA IN 1968. Below is a snippet that is very on point:


"GOTO Considered Harmful" Considered Harmful
The most-noted item ever published in *Communications* was a letter from Edsger W. Dijkstra entitled "Go To Statement Considered Harmful" [1] which attempted to give a reason why the **GOTO** statement might be harmful. Although the argument was academic and unconvincing, its title seems to have become fixed in the mind of every programming manager and methodologist. Consequently, the notion that the **GOTO** is harmful is accepted almost universally, without question or doubt. To many people, "structured programming" and "**GOTO**-less programming" have become synonymous.

Although the argument was academic and unconvincing, its title seems to have become fixed in the mind of every programming manager and methodologist. Consequently, the notion that the GOT0 is harmful is accepted almost universally, without question or doubt. To many people, "structured programming" and "GOTO-less programming" have become synonymous.

This has caused incalculable harm to the field of programming, which has lost an efficacious tool. It is like butchers banning knives because workers sometimes cut themselves. Programmers must devise ellaborate workarounds, use extra flags, nest statements excessively, or use gratuitous subroutines. The result is that GOTO-less programs are harder and costlier to create, test, and modify.

SWITCHING TO THE OSP STUDIO, OCTOBER 2023, WHERE I AM MEETING DORIANE TO SPEAK A BIT MORE ABOUT MY DEEP DIVES INTO THE CSS REGIONS.

d: HTML is text and text is linear. If you look at one HTML file, you have one line after another. If you want to break the content into multiple parts, the break has to happen between two lines. And that is how page breaks and column breaks happen in a HTML file.

m: But what kind of break it is, is still up to the designer right?

d: Yes there is page break, column break, and i think there is a third one.

m: I think region break is still there, but it is a legacy property.

d: Yes oke. It's like we have a linear flow and between two lines we can say there is a page break, column break or region break. What i want to say, to answer what you said about that it's crazy that Paged.js does not accept two flows... How do you accept two flows in the linearity of a document? The document is linear and the break has to be inserted in this linearity. So, yes, the way you will have to flow is to use this paradoxical confusion of saying...

m: So you're thinking from the understanding of a HTML document. Which are linear, from top to bottom, and they don't have notions about placing things side by side, if you think from the point of view of the semantics of HTML. But if you think from the practice of book design, and reading, then it's very possible, no? To have to things on one page, or even a sort of horizontal flow that continues. And these two practices, the web practices and design practices are coming together in web-to-print, but it stays very stuck in a single flow way of thinking about a page.

d: I agree. This linearity comes from the linearity that we write HTML line by line.

I can see why it causes trouble.

m: I find it quite a big statement to say, that it's not possible to implement something like that at all, in a way that does not go against general ways of working with HTML and CSS. Because in a way, what Paged.js is doing, it also fragments your document into pages. There is a chunker at work and it's making decisions about where to place a page break. I'm just wondering why a page break should stay a singular one. To me, as a designer, it feels weird to just accept the logic of the flow, where HTML is based on, and fully give up on layouts based on multiple flows. Because the CSS Regions are an example to show that there are ways to do this.

d: Yes, there are ways, it works. But those ways are going against the ideology of HTML and CSS that were proposed at the start. And this is something super conservative to say also. Like, oh no we cannot use it like that because it was not meant to be...

m: Which would be sad.

d: Yes that would be sad. So I'm totally for the hacks and to break the material and make something else out of it. But, it's still important to see this kind of approach for what it is, as a hack craft that is part of DIY culture. And I see that ideologically it cannot be part of standard culture. Because it breaks the very... Well, it's not harmful, it is wrong to say that it is harmful, but it is true to say, to me, that CSS Regions break the main ideology of HTML and CSS.

m: And if there would be another way?

d: But then, what? :)

m: A way of working in which you can keep your content and place them, a bit like the CSS grid system but then with flows?

d: But to me, my instinct is that this theoretical new way of doing regions... I don't believe a lot in it. We're speaking about something that does not exist yet, so yes there is the possibility to invent something there. But my instinct is that because of the at one side the linearity of a HTML document and on the other side a multiple flow that needs to break on multiple positions, you will have to make a bridge between those two, in a way that is respectful to the conservative standard let's say. And to me making such a bridge while being respectful to the conservative standard is mission impossible. I can be surprised of course!

m: I'm nicely surprised by your counter opinion.

d: That I'm not deeply in favor of CSS Regions?

m: Yes yes. Because it seems also related to your respect to the ideology. And your preference to stay with it. I'm just thinking out loud now... If you think about CSS and HTML and the ideology as something to follow, you have to do with a lot of people, no?, that you don't want to confuse. And if you take a hack/craft approach, you are only confuse yourself. It's a very different situation, the stakes are very different.

On the ideological side, if you say: I wouldn't personally be in favor to bend the ideology that much, for designers to get more space on the web to make more interesting layouts... I'm just wondering, can you say a bit more about that? There is something really interesting going on here I think.

d: Yes. But it's really a matter of personal opinion! The first difference that you're pointing out is that you're either you make something based on the standard and your public is the large community that is using it. Or you make a craft and it's a niche. But it's not really about that, it's really about taste I think.

To me it is not so interesting to make editions out of the web, if it is not to reflect a bit on the materiality of the web.

m: With "edititions" you mean publications and books?

d: Yes. Printed matter.

If web-to-print is becoming this kind of counter culture against the hegemony of Adobe's proprietary software, etc, and we will have new tools to do design work, I think it's super cool. But then, it's also nice to see that this counter-culture of web-to-print, comes with certain aesthetics. In the same way that there is a punk aesthetic. Punk culture has a certain aesthetic tight to it, and it's for a reason. It's because they use certain material and come from a certain background. And because of history, then those aesthetical choices reflect political or ideological opinion, on how we should do things. And to me this link is interesting.

So with all that said... To bend HTML and CSS to make it look more like something that does not look like HTML and CSS, is not that important to me. If I'm printing something out of HTML and CSS, I want that it's visible that it comes from there, because then the aesthetic also speak. Then people ask questions like, oh why does it look like that way? Oh it is designed in that way, why is it designed in that way? Oh it is because you are against using that kind of software, why are you against it? And then you start a conversation.

But if the book looks like any Gutenberg book design from the last 100 years, then I don't see why we're doing that. I like the idea that it's not only about the tools but also about the culture that is created out of it. And then there is this idea that doing web-to-print has to be close to the medium of the web, or the materiality of the web, but this is not only about linear gradients and border styles, it's also about global composition.

And to use a hacked browser with something that goes against the ideology of the material... Even if the arguments why going against the ideology are conservative, and I don't want to agree with those arguments, because I would like to let people use the tools as they want to, this is fine. If I want to be close to the material, and put up front this new culture of printed documents, I also want to be faithful to the material.

Maybe we can summarize it by saying that the desire to be close ideologically to certain tools or materials, can have two sides: the conservative side and the punk aesthetic side. I want to show the tool, but also then I'm also trying to be super faithful to the default aesthetics of the tools can also become quite conservative in a way. And those two sides need to be in balance a bit.

So to me, it's not about the layout tools being used by a large community or a small community of hackers, it's not really about that, it's about what we say aesthetically and culturally to the people who will read those books. At which point do we want to show them that, yes, it's made from a web document, and yes, web documents work in this specific way, so no this is not possible to do and I'm not going to bend reality just to make it more academic to you. I'm accepting how HTML and CSS works, as it is, without being critical at anything, because this is how it works.

m: Last question then, and this is a big one again, but how do you feel about the position of the W3C and the formulation of web standards and the people who are in those work groups, that represent the biggest companies that use the internet and thus have the power to say what will be in- and outside of the HTML and CSS standards. This is a hegemony as well, that you have to deal with when you work with the web. So I'm curious how you feel about working within that environment?

d: Yes… I have all those ideas about web-to-print as an aesthetic and how it should be put forward, in my opinion. But those kind of aesthetics are also context sensitive. What do you want to do? Do you want to do a small punk zine, or do you want to replace a printing system of a large industry? But really, the printing system of a large industry also has to be changed into something open source and made out of the web, and not something punk. But in my designer position, I'm more often in the position of weird artistic zines.

m: And you see space for those zines within an environment of the web, that is controlled by the hegemony of the web?

d: Yes, I don't have much knowledge on this. But punk and zine culture, alternative culture of printed documents, or folklore… have always found themselves in the margins of using something that is very corporate and mainstream, by appropating the technology for themselves. Zines are made with Xerox photocopiers, does this mean that the punk scene agrees with the Xerox ideology? No. But they did inherit the materiality of it. In the way that those machines work.

In a way I could see the small web-to-print editions a bit in the same way. It's not that you're giving thumbs up to the W3C, but that's what we have. And it is right now something that is easily usable by a lot of people that we can play with, so let's play with it. And let's show its aesthetics in a raw way. Like photocopies in old zines coming out of a Xerox machine, or something like that. Or risograph and all those technologies. And it's something else to do the job of talking to Xerox or RISO to say that they should change their machines. No the punk position is much more like saying, oke we have a Xerox at my place and we can just use it to print zines. And for now, my personal position is much more the punk folkloric one. I can use this and show its aesthetics.

And so I don't have much of an opinion about the W3C… Well yes I can have an opinion about a specific topic, of course, but I don't have much to say about the working groups and such, because I don't want to be part of that discourse. I just want to use their technology, or something that makes sense to me.

Yes, I don't know. Maybe it would be interesting to enter the discussion with the working groups… But really what interests me is to be close to the material, the vernacular, folkloric and punk side of editions… This is why web-to-print is exciting to me right now. But if let's say, we will have discussions with those working groups, and in 5 years you can easily do printed matter on the web, and big corporations are starting to produce print on the web, and web-to-print becomes like a mainstream service, it will be appropriated by capitalism and it will be probably horrible. You probably will need to use a lot of extensions and ready made software to do it. And I know that I would be totally desinterested in it. I would just switch to the new punk way and alternative way to do things. And of course in this new way of doing things, a lot of things will be fragile and not work, and this is exactly why it's nice.

I had this discussion with Gijs about Paged.js, about the fact that it is a polyfill and that at some point you can remove the polyfill and everything will just work… At this point I will probably stop doing web-to-print.

m: Ah yes, then already?

d: Already? I don’t know. It would mean that certain actors will be very interested in making printed matter with the web. And they are not going to be interested in making it work because some niche designers, there will be big plans behind it.

m: So it’s very context sensitive for you?

d: Yes it is very context sensitive. And at this point we will go back to make publications with Python, or whatever. But to me, it’s not only about allowing something technically or not, and fighting for something to be allowed technically or not, it’s also about alternative culture and folklore. And this is what interests me. Small spaces with oral pedagogy, and learning from your peers, and doing things collectively together… And this comes with a whole cultural context. Just making tools accessible, if it’s to be able to start working like a startup, I don’t see the point.

m: It’s super interesting to hear your position. So in the end you say it’s a personal preference where you position yourself, in a way you’re drafting out a map and you say, oke, I’m here and here.

d: Yes yes.

m: You make a clear cut for yourself to say, oke, I don’t want to change HTML, I want to take it as it is. Where I think for me, the excitement about the CSS Regions comes from the possibility to not fully embrace HTML as one paradigm that just continues and follows the logic of updates all the time. And the whole thinking that you can’t go back in time is weird to me, because you actually can go back to a certain browser that still supports the CSS Regions, even though they were just implemented for just a very short time. I think it’s super interesting that you can move against the grain of those development driven and future forward ways of thinking, in terms of adjusting how things work. For that is super interesting.

So for myself, I find this as a hack/craft approach super interesting, because it allows me to think about why things change as they change. And to learn about it, and to read discussions of the W3C, and to see what kind of opinions are being formulated there, and to learn from it and formulate my opinions in the mean time.

So yes, it’s super helpful to have a graph like this, to orient yourself within this world. It makes it very exciting. It makes it exciting to disagree and to take different approaches.

d: I don’t even think it’s a disagreement. I also agree on the hack/craft approach, I think it’s wonderful to bend a software into something that it is not meant to do in the first place, and to say whataver, let’s go against the ideology of the standard. I think it’s also super cool. It’s just that I am not the person who is going to put all my energy into doing that, but I totally support that. I think both are important.

m: In the end for me it’s not even about the tool, to make sure for example that OSPKit is available for another 20 years, it’s also a lot about the thinking with it, and understanding, and learning. This is for me the motivation to do this work and to document it.

d: Can I add something? Because now on this graph we have on one side being faithful to HTML and CSS, and on the other side to do hack/craft by breaking it.

m: Or to not follow the modern up-to-date regimes..

d: Yes. Allright, but there is something that is not there on the graph right now. Because there are two different things: it’s about doing hack and craft for yourself or your small community, by bending the software and go beyond, but you also have those forces who are trying to turn the hack and craft approach back into the new ideology of the standard again. I can support the hack and craft and all of that, but this arrow is another question.

I can situate practices that just work with HTML and CSS, that work without regions breaks, there is no polyfill, etc. Or I can do something with the Paged.js polyfill, or OSPKit and CSS Regions, and have a lot of fun. But then to say, we should put back CSS Regions in the standards, or we should put the Paged.js polyfill back into the standard, it will remove these practices from their hack and craft cultural context.

The thing with Paged.js is that it is faithful to HTML and CSS ideology, but also faithful to a hegemonic cultural context.

m: Yes it fits the needs of the large print industry.

d: Yes which is something different then making a punk scene out of HTML and CSS, without any hacking or crafting, just accepting anything that comes from the browser. Without trying to bend the tools.

m: Yes. There is something interesting that here, with the punk side, there is a desire to stay close to form and aesthetics, and intention, and the question for who the publication is is an important aspect for making it in a punk way. Where here, in the large print industry, there is a disconnect between tool and aesthetics, as it does not really matter if a book is made in HTML. Where here, in the hack/craft approach, form and content and tool and aesthetics come very close to each other, as it is all about zooming in and staying very close to the materiality of both tools, content and context.

d: Maybe to situate a bit more my opinion, and I'm a bit thinking out loud, my personal opinion about that comes from a personal experience, as I did not have studied as a designer, but as an artist. And this is an idea that is really rooted in artistic practice, the idea that you should embrace the medium. If you are working with ceramics or food or with whatever, you should take advantage of this medium, you should show the grain and not try to disguise it. Art can still disguise it of course, but this is an art school philosophy. When doing net art for example, you should be concious of the environment. Or when making work for a white room, you will have to be concious of that context.

This very much comes from an art perspective on it. I can imagine that a designer would not think as much from the origins of the material but would think a lot about usage, and how people receive it, and what impact it creates to have the text shaped in a certain way. Where I think a lot about the materiality of it.

m: I understand, but in the example of net art, I find it a pity that sometimes the whole organisational procedures and the politics behind an environment are not seen as the material, that also shape the environment in the end. There is a focus on the aesthetics and materiality, but I just find it so interesting to also look further then that, and to understand who is deciding, who says that it should be like that, and what is happening in the mean time. I think this is an interesting layer to include when thinking about web-to-print, because this makes it more than an aesthetic or material. I'm not saying that I don't find aesthetics and materiality important, on the contrary. Aesthetics can speak to a big crowd, it's a super powerful part of it. But I'm also interested in looking further, and then I bump into the CSS Regions story and everything that happens behind it. And it makes me interested in understanding these procedures of how growth is defined within the context of the web, and how you can go against that, you don't have to follow it. It is another way to work with what the web is for me, that is maybe not depending on the aesthetic side of it.

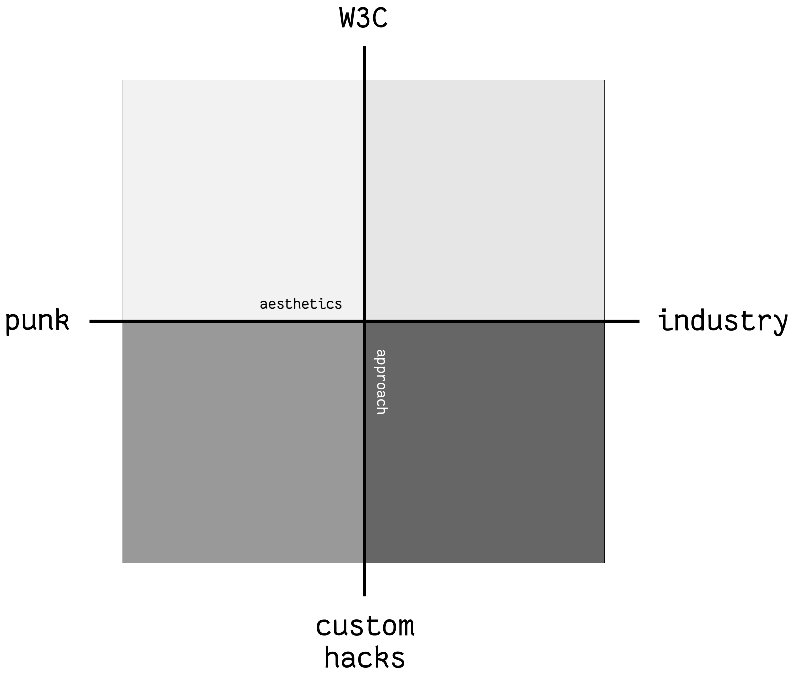
d: But if you don't communicate the process of your craft through the aesthetics of it, then you will have to communicate it in another way.

m: That is a good point, yes.

d: To me this is a bit the problem with Médor, there are no traces of HTML and CSS in it. To me it's sad, it's really sad. I would like that people open Médor and that people feel that it is a bit like a website. Even if we use the CSS Regions, you can be crafty and go against the ideology of the standard, but still it's important to show this craft in a certain way. Should we just draw an arrow between the boxes? Or should we have an external flow again somewhere else that is linear? There is a lot of play that can in some way reveal this craft, that I would find interesting.

m: Yes it is a super good point, how do you reveal all the interest in the background processes. It's so hard to make them visible and let them speak in layout. There must be other ways next to aesthetics.

d: Yes there are other ways. But still I have something super I would say child like. When I see something is printed with a border-radius 50%, you can directly see that it is that, because you have a lot of element of different sizes and the ellipses are spaces differently in each object. And this is super nice to see, because it comes from certain material, and you see that without any kind of word of



web-to-print political compass

conversation. I'm not understanding aesthetic as something elegant or whatever, but as something that speaks. Sometimes just recognizing one CSS property is just wow.

So in a way my question is, oke, nice to do hacky CSS Regions craft, but how can you express this process in a similar way? If you don't go into the direction of following the standard and you go into the direction of bending the standard, then how can you show those cultural elements that are linked to this craft within the final object? How can you express it in the object that you're printing?

This zine is made with HTML and CSS, using OSPKit and ✨ CSS Regions ✨.
pdfbook2 is used to turn the PDF into a booklet PDF.

The title is a version of "Dancing in the ruins of big tech", one of the slogans of the digital depletion strike. https://titipi.org/wiki/index.php/Slogans

Pre-publishing moment on the occassion of zine camp 2023, Rotterdam.

With many thanks to OSP for my residency period.