post0

Charles Baker

October 10, 2025

Contents

NOTE #+OPTIONS: H:6 makes the exporter create outline-containers beyond the default 4 levels (this is needed for folding)

1 Progress bar

2 Features to make building easier

- custom exporter for plotly? instead of having to manually insert the iframe? maybe a python function could take care of this

3 narrative

3.1 link to CSS and JSS scripts

3.2 Long paragraph

lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim[1] id est laborum. Sed ut perspiciatis unde omnis iste natus error sit voluptatem accusantium doloremque laudantium, totam rem aperiam, eaque ipsa quae ab illo inventore veritatis et quasi architecto beatae vitae dicta sunt explicabo. Nemo enim ipsam voluptatem quia voluptas sit aspernatur aut odit aut fugit, sed quia consequuntur magni dolores eos qui ratione voluptatem sequi nesciunt. Neque porro quisquam est, qui dolorem ipsum quia dolor sit amet, consectetur, adipisci velit, sed quia non numquam eius modi tempora incidunt ut labore et dolore magnam aliquam quaerat voluptatem. Ut enim ad minima veniam, quis nostrum exercitationem ullam corporis suscipit laboriosam, nisi ut aliquid ex ea commodi consequatur? Quis autem vel eum iure reprehenderit qui in ea voluptate velit esse quam nihil molestiae consequatur, vel illum qui dolorem eum fugiat quo voluptas nulla pariatur?

3.3 TODO prompt LLM (is there an org gptel or something similar?)

3.4 layout

3.4.1 css grid

TODO get syntax highlighting working

[1] new footnote

3.5 internal links

Post 1

3.6 global header / site style

3.7 DONE styling

3.7.1 DONE inline css

3.7.2 DONE org-html-themes

`https://github.com/fniessen/org-html-themes` `https://github.com/fniessen/org-html-themes/blob/master/examples/org-mode-syntax-example.org`
(using read the docs see above)

3.7.3 DONE css (see above)

DONE global css

DONE local css

3.8 TODO basic org stuff

3.9 TODO text

3.9.1 long paragraph

3.9.2 long paragraph scrolling text

3.9.3 quote

this is some regular text

> This is some quote

this is some more regular text

3.10 tags TAG0:TAG1

3.10.1 DONE test todo item 2

3.10.2 DONE table

3.10.3 TODO lists

TODO bullets NOTE: these render as numbers

a	b	c
1	2	3
4	5	6

- one
- two
  - two point one
  - two point two
- three
  - three point one
  - three point two

TODO numbers

1. first
2. second
   (a) second point one
   (b) second point two

3.10.4 TODO formatting

- some **bold** text
- some *italic* text
- some underline text
- some ~~strike~~ text
- some `code` text
- some `verbatim` text
- some `*bold in code*` text
- some `/italic in verbatim/` text
- some `_underline in verbatim_` text

- some =~code in verbatim=~ text
- some `+strike in verbatim+` text
- some **bold** *italic* underline ~~strike~~ `code` `verbatim` text

3.11 TODO code

3.11.1 TODO bash

```
echo "hello world"
```

```
hello world
```

3.11.2 TODO python

- NOTE: need to set up dir-locals[2]

TODO using venv This is how you write code with python[3]

```
from dataclasses import dataclass

from polyfactory.factories import DataclassFactory


@dataclass
class Person:
    name: str
    age: float
    height: float
    weight: float


class PersonFactory(DataclassFactory[Person]):
    ...
```

[2] lorem impsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum

[3] This is how to write code with python

```
person_instance = PersonFactory.build()
print(person_instance)

print('hello')
```

```
Person(name='pWXqLGtQKLbTIJjpCure', age=-92.811375428167, height=7.15080271143585, weig
hello
```

TODO using multiple venvs via session

TODO using session to pass state between blocks

3.12 TODO video (test with git lfs)

3.13 TODO image

3.13.1 DONE local file

3.13.2 TODO remote file

3.13.3 TODO svg

3.14 DONE general iframe

3.15 DONE plotly chart

```
import plotly.express as px
import pandas as pd
```

```
df = pd.DataFrame({
    "Fruit": ["Apples", "Oranges", "Bananas", "Apples", "Oranges", "Bananas"],
    "Amount": [4, 1, 2, 2, 4, 5],
    "City": ["SF", "SF", "SF", "Montreal", "Montreal", "Montreal"]
})

fig = px.bar(df, x="Fruit", y="Amount", color="City", barmode="group")

# write to html file
fig.write_html("html/my_interactive_plot.html")
```

insert iframe to chart

3.16 DONE mermaid diagrams

[width=.9]images/test-diagram

3.17 TODO verbs

3.18 TODO footnotes NOTE issue

This is a footnote[4]. And this is some other text

3.18.1 Another footnote

More[5] footnotes[6].

[4]This is the footnote. lorem impsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. **Excepteur sint occaecat cupidatat** non proident, sunt in culpa qui officia deserunt mollit anim id est laborum. Sed ut perspiciatis unde omnis iste natus error sit voluptatem accusantium doloremque laudantium, totam rem aperiam, eaque ipsa quae ab illo inventore veritatis et quasi architecto beatae vitae dicta sunt explicabo. Nemo enim ipsam voluptatem quia voluptas sit aspernatur aut odit aut fugit, sed quia consequuntur magni dolores eos qui ratione voluptatem sequi nesciunt. Neque porro quisquam est, qui dolorem ipsum quia dolor sit amet, consectetur, adipisci velit, sed quia non numquam eius modi tempora incidunt ut labore et dolore magnam aliquam quaerat voluptatem. Ut enim ad minima veniam, quis nostrum exercitationem ullam corporis suscipit laboriosam, nisi ut aliquid ex ea commodi consequatur? Quis autem vel eum iure reprehenderit qui in ea voluptate velit esse quam nihil molestiae consequatur, vel illum qui dolorem eum fugiat quo voluptas nulla pariatur?

[5]even more

[6]More footnotes

3.19 TODO transclude? compose with other file?

3.19.1 include from another file

3.20 TODO arrange two views next to eachother (eg figure and some figure text) using css grid (might be split between here and css). is ther a way to include the css here? can't we tangle to a file

3.21 How this site is built

3.21.1 minimap

CSS

```
/* Minimap styles */
#minimap-container {
  position: fixed;
  bottom: 0%;
  right: 0%;
  width: 6%;
  height: auto;
  max-height: 100%;
  z-index: 10000;
  pointer-events: none;
  /* need auto otherwise we can't scroll the minimap independently */
  overflow: auto;
  filter: blur(0.1px) grayscale(90%);
  margin: 1rem;
}

/* NOTE overflow hidden works, but it causes the main page to scroll for some reason */
#minimap {
  position: relative;
  overflow: auto;
  pointer-events: auto;
  box-sizing: border-box;
}

#minimap-content {
  transform-origin: top left;
  pointer-events: none;
```

```
}

#minimap-viewport {
  position: absolute;
  box-sizing: border-box;
  z-index: 10;
  background: rgba(0,0,0, 0.1);
  pointer-events: none;
}
```

Javascript

```
// minimap
function minimap_update(called) {
    // Parameters
    const minimapWidth = document.body.clientWidth * 0.1;
    const scale = minimapWidth / document.querySelector("#content").scrollWidth;

    // Setup the minimap
    const minimap = document.getElementById("minimap");
    let minimapContent = document.createElement("div");
    minimapContent.id = "minimap-content";
    minimap.appendChild(minimapContent);

    // Clone body content (shallow, not perfect for all apps)
    function cloneBodyContent() {
        const clone = document.querySelector(‘#content‘).cloneNode(true);
        // change id of the clone node to content-clone
        clone.id = "content-clone";
        // Remove the table of contents from the clone
        const toc = clone.querySelector("#table-of-contents");
        if (toc) toc.remove();
        // Remove the footnotes from the clone
        const fn = clone.querySelector("#footnotes");
        if (fn) fn.remove();
        // Remove the copilot summary from the clone
        const tldr = clone.querySelector("#text-tldr");
        if (tldr) tldr.remove();
        // set the margins of the clone to 0
        clone.style.margin = "0";
```

```
    // Remove the minimap itself from the clone
    const mmc = clone.querySelector("#minimap-container");
    if (mmc) mmc.remove();
    minimapContent.innerHTML = "";
    minimapContent.appendChild(clone);
    minimapContent.style.transform = `scale(${scale})`;
    minimapContent.style.width = document.body.scrollWidth + "px";
    minimapContent.style.height = document.body.scrollHeight + "px";
}

// Viewport rectangle
const viewport = document.getElementById("minimap-viewport");

function updateViewport() {
    const bodyScale =
        (document.body.clientWidth * 0.1) / document.body.scrollWidth;
    const scrollTop = window.scrollY;
    const scrollLeft = window.scrollX;
    const viewportWidth = window.innerWidth;
    const viewportHeight = window.innerHeight;

    viewport.style.top = scrollTop * bodyScale + "px";
    viewport.style.left = scrollLeft * bodyScale + "px";
    viewport.style.width = viewportWidth * bodyScale + "px";
    viewport.style.height = viewportHeight * bodyScale + "px";
}
window.addEventListener("scroll", updateViewport);

// Style fired up on window resize (and periodically for dynamic content)
function updateContentScale() {
    const realScale =
        (document.body.clientWidth * 0.1) / document.body.scrollWidth;
    minimapContent.style.transform = `scale(${realScale})`;
    minimapContent.style.width = document.body.scrollWidth + "px";
    minimapContent.style.height = document.body.scrollHeight + "px";
    updateViewport();
}

addEventListener("resize", updateContentScale);
```

```
    cloneBodyContent();
    updateContentScale();

    // Clicking on minimap scrolls page
    minimap.addEventListener("click", function (e) {
        const minimapWidth = document.body.clientWidth * 0.1; // 20% of body width
        const scale = minimapWidth / document.body.clientWidth;

        const minimap = document.getElementById("minimap");
        const rect = minimap.getBoundingClientRect();
        const x = e.clientX - rect.left;
        const y = e.clientY - rect.top;

        const scrollX = x / scale;
        const scrollY = y / scale;

        window.scrollTo({
            top: scrollY - window.innerHeight / 2,
            left: scrollX - window.innerWidth / 2,
            behavior: "smooth",
            block: "nearest",
        });
    });

    //// Make unobtrusive
    //minimapContainer = document.getElementById("minimap-container");
}

document.addEventListener("DOMContentLoaded", function () {
    // Example: access and manipulate an HTML element
    minimap_update(false);
});

// includes window resizing as well as text scale increase and
// decrease
window.addEventListener("resize", function () {
    // Example: access and manipulate an HTML element
    console.log("resize event detected");
    minimap_update(true);
});
```

```
// autoscroll minimap TODO fix this as I think the get attribute by id
// is ambigious and just happens to return the correct one
window.addEventListener("DOMContentLoaded", () => {
    const observer = new IntersectionObserver((entries) => {
        entries.forEach((entry) => {
            // if entry is a child of #minimap ignore it
            if (entry.target.closest("#outline-container-tldr")) {
                return;
            }
            // if entry is a child of #minimap ignore it
            if (entry.target.closest("#minimap-container")) {
                return;
            }
            let id = "";
            // Check if target is an outline-text div
            id = entry.target.getAttribute("id");

            id = CSS.escape(id);
            const elementLink = document.querySelector(`#${id}`);
            if (elementLink) {
                if (entry.intersectionRatio > 0) {
                    elementLink.scrollIntoView({
                        behavior: "smooth",
                        block: "center",
                    });
                }
            }
        });
    });

    // Collect all headings and outline divs
    const headings = [...document.querySelectorAll("#content *")];

    headings.forEach((heading) => observer.observe(heading));
});
```

3.21.2 progress-bar

CSS

```
/* progress bar */
#progressBar {
    position: fixed;
    top: 0;
    left: 0;
    height: 5px; /* Adjust thickness as needed */
    background-color: gray; /* Progress bar color */
    width: 0%; /* Initial width is 0% */
    z-index: 9999; /* Ensure it stays on top */
}
```

Javascript

```
// progress bar
// script.js
document.addEventListener("DOMContentLoaded", () => {
    const progressBar = document.getElementById("progressBar");

    const updateProgressBar = () => {
        const scrollTop =
            document.documentElement.scrollTop || document.body.scrollTop;
        const scrollHeight =
            document.documentElement.scrollHeight -
            document.documentElement.clientHeight;
        const scrollPercentage = (scrollTop / scrollHeight) * 100;

        progressBar.style.width = `${scrollPercentage}%`;
    };

    // Initial call to set progress bar on page load
    updateProgressBar();

    // Add scroll event listener to update the progress bar
    window.addEventListener("scroll", updateProgressBar);
});
```

3.21.3 footnotes

CSS

```
/** sticky footnotes **/
```

```
#footnotes {
    position: fixed; /* Fixes the div to the viewport */
    bottom: 2% ;   /* Distance from the bottom */
    left: 0%;    /* Distance from the right */
    width: 20%;   /* Set a width for the TOC */
    overflow-y: auto; /* Enables vertical scrolling */
    overflow-x: hidden; /* Hide horizontal scrollbar if any */
    max-height: 42%; /* Ensures it doesn't exceed viewport height */
    margin: 1rem;
}
#footnotes h2 {margin: 0;}

#footnotes::-webkit-scrollbar {display: none;}

#text-footnotes a, #text-footnotes p {
  text-decoration: none;
  padding: .125rem 0;
  color: #ccc;
  /* smooth */
  transition: all 50ms ease-in-out;
}

#text-footnotes a:hover,
#text-footnotes a:focus {color: #666;}
#text-footnotes p:hover,
#text-footnotes p:focus {color: #666;}

#text-footnotes ul, #text-footnotes ol {
  list-style: none;
  margin: 0;
  padding: 0;
}

#text-footnotes p.footpara.active {color: #333; font-weight: bold;}
#text-footnotes sup a.active {color: blue;}
```

Javascript

```
// Followable, scrollable footnotes
window.addEventListener("DOMContentLoaded", () => {
```

```
    const observer = new IntersectionObserver((entries) => {
        entries.forEach((entry) => {
            // ignore if entry is child of #content-clone
            if (entry.target.closest("#content-clone")) {
                return;
            }
            // Check if target is an outline-text div
            id = entry.target.getAttribute("id");

            const fnLink = document
                .querySelector(`#footnotes sup a[href="#${id}"]`)
                .closest("sup")
                .nextElementSibling.querySelector("p");
            const fnLink1 = document.querySelector(
                `#footnotes sup a[href="#${id}"]`,
            );
            if (fnLink) {
                const action = entry.intersectionRatio > 0 ? "add" : "remove";
                fnLink.classList[action]("active");
                fnLink1.classList[action]("active");

                // Scroll active link into view
                if (entry.intersectionRatio > 0) {
                    fnLink.scrollIntoView({
                        behavior: "smooth",
                        block: "nearest",
                    });
                }
            }
        });
    });

    // Collect all headings and outline divs
    const headings = [...document.querySelectorAll("sup a.footref")];

    headings.forEach((heading) => observer.observe(heading));
});

// Clickable Footnotes
document.addEventListener("DOMContentLoaded", function () {
```

```
    const tocLinks = document.querySelectorAll("#footnotes a[href^='#']");

    tocLinks.forEach((link) => {
        link.addEventListener("click", function (event) {
            event.preventDefault();
            targetId = this.getAttribute("href").substring(1);
            targetId = CSS.escape(targetId);
            const elements = document.querySelectorAll(`#${targetId}`);
            const targetElement = Array.from(elements).filter(
                (el) => !el.closest("#content-clone"),
            )[0];
            if (targetElement) {
                targetElement.scrollIntoView({
                    behavior: "smooth",
                    block: "start",
                });
            }
        });
    });
});
```

3.21.4 foldable-headings

CSS

```
.outline-6 {cursor: pointer; }
.outline-6.folded {max-height: 2em; background: #f9f9f9; overflow: clip;}

.outline-5 {cursor: pointer; }
.outline-5.folded {max-height: 2em; background: #f9f9f9; overflow: clip;}

.outline-4 {cursor: pointer; }
.outline-4.folded {max-height: 2em; background: #f9f9f9; overflow: clip;}

.outline-3 {cursor: pointer; }
.outline-3.folded {max-height: 2em; background: #f9f9f9; overflow: clip;}

.outline-2 {cursor: pointer; }
.outline-2.folded {max-height: 2em; background: #f9f9f9; overflow: clip;}

#outline-2::-webkit-scrollbar {display: none;}
```

```
#outline-3::-webkit-scrollbar {display: none;}
#outline-4::-webkit-scrollbar {display: none;}
#outline-5::-webkit-scrollbar {display: none;}
#outline-6::-webkit-scrollbar {display: none;}
```

Javascript

```
// Foldable divs
document.addEventListener("DOMContentLoaded", function () {
    const foldableDivs = document.querySelectorAll('[class^="outline-"]');

    foldableDivs.forEach((div) => {
        div.addEventListener("click", function () {
            event.stopPropagation(); // Prevents the event from bubbling up to parent c
            this.classList.toggle("folded");
        });
    });
});
```

3.21.5 toc

CSS

```
/** sticky ToC **/
#table-of-contents {
    position: fixed; /* Fixes the div to the viewport */
    top: 1%;      /* Distance from the top */
    bottom: 2% ;   /* Distance from the bottom */
    left: 0%;    /* Distance from the right */
    width: 20%;   /* Set a width for the TOC */
    overflow-y: auto; /* Enables vertical scrolling */
    overflow-x: hidden; /* Hide horizontal scrollbar if any */
    max-height: 50%; /* Ensures it doesn't exceed viewport height */
        margin: 1rem;

}

#table-of-contents h2 {margin: 0;}

#table-of-contents::-webkit-scrollbar {display: none;}
```

```
#text-table-of-contents a {
  text-decoration: none;
  display: block;
  //padding: .125rem 0;
  color: #ccc;
  /* smooth */
  transition: all 50ms ease-in-out;
}

#text-table-of-contents a:hover,
#text-table-of-contents a:focus {color: #666;}

#table-of-contents ul, #table-of-contents ol {
  list-style: none;
  margin: 0;
  padding: 0;
}

#text-table-of-contents li.active > a {
  color: #333;
  font-weight: bold;
}
```

Javascript

```
// Followable, scrollable table of contents
window.addEventListener("DOMContentLoaded", () => {
    const observer = new IntersectionObserver((entries) => {
        entries.forEach((entry) => {
            if (entry.target.getAttribute("id") === "text-tldr") {
                return;
            }
            // ignore if entry is child of #content-clone
            if (entry.target.closest("#content-clone")) {
                return;
            }
            let id = "";
            // empty headings
            if (entry.target.tagName.match(/^H[2-6]$/)) {
                id = entry.target.getAttribute("id") || "";
```

```
    const tocLink = document.querySelector(
        `#text-table-of-contents a[href="#${id}"]`,
    );
    const nextSibling = entry.target.nextElementSibling;
    if (tocLink && !nextSibling) {
        const action =
            entry.intersectionRatio > 0 ? "add" : "remove";
        tocLink.parentElement.classList[action]("active");

        // Scroll active link into view
        if (entry.intersectionRatio > 0) {
            tocLink.scrollIntoView({
                behavior: "smooth",
                block: "nearest",
            });
        }
    }
}

// Check if target is an outline-text div
if (
    entry.target.tagName === "DIV" &&
    entry.target.className.match(/outline-text-[2-6]/)
) {
    id =
        entry.target.previousElementSibling?.getAttribute("id") ||
        "";

    const tocLink = document.querySelector(
        `#text-table-of-contents a[href="#${id}"]`,
    );
    if (tocLink) {
        const action =
            entry.intersectionRatio > 0 ? "add" : "remove";
        tocLink.parentElement.classList[action]("active");

        // Scroll active link into view
        if (entry.intersectionRatio > 0) {
            tocLink.scrollIntoView({
```

```
                            behavior: "smooth",
                            block: "nearest",
                        });
                    }
                }
            }
        });
    });

    // Collect all headings and outline divs
    const headings = [
        ...document.querySelectorAll("h2[id], h3[id], h4[id], h5[id], h6[id]"),
        ...document.querySelectorAll(
            "div.outline-text-2, div.outline-text-3, div.outline-text-4, div.outline-te
        ),
    ];

    headings.forEach((heading) => observer.observe(heading));
});

// Clickable ToC
document.addEventListener("DOMContentLoaded", function () {
    const tocLinks = document.querySelectorAll(
        "#text-table-of-contents a[href^='#']",
    );

    tocLinks.forEach((link) => {
        link.addEventListener("click", function (event) {
            event.preventDefault();
            const targetId = this.getAttribute("href").substring(1);
            const elements = document.querySelectorAll(`#${targetId}`);
            const targetElement = Array.from(elements).filter(
                (el) => !el.closest("#content-clone"),
            )[0];
            if (targetElement) {
                targetElement.scrollIntoView({
                    behavior: "smooth",
                    block: "start",
                });
            }
```

```
        });
    });
});
```

4 Generate a TLDR for this document that will be displayed at the top of the static-site version of this document. Include internal org links to headings in this document where relevant. Org links look like this: ??. Do not use any bullets for the explanation, but you can use line breaks to help with readability. Be verbose and describe as many sections as possible

This document serves as a comprehensive demonstration of Org mode authoring for static sites, showing advanced features and customizations. At the top, a progress bar and minimap enhance navigation. The document explores embedding interactive charts with Plotly, and suggests automations for easier iframing. It includes narrative text, CSS and JS linking details, and example paragraphs, quotes, and layout concepts.

There's an overview of using both modern layouts via CSS grid, with practical HTML examples. Internal links (such as Post 1) and site-wide style are covered, including the use of org-html-themes, local and global CSS, and custom inline styles.

Basic Org constructs are featured under 3.8, including tagged sections, lists and formatting, and tables. Source code examples are provided for bash scripts and python, including venv and session hints. Embedding media is demonstrated with images, remote files, SVG diagrams, and eg Mermaid charts.

There's also coverage of internal verbatim and code syntax, proper handling of footnotes with references, and ideas for future transclusion and file composition.

Document concludes with a detailed footnotes section and plans for CSS-driven layout enhancements, including arranging multiple views or panels side by side (figures + captions) using CSS grid.