Instruct

Author: Yuri Tricys
Date: August 10th, 2025

PIDP:3240

Media Enhanced Learning

LESSON PLAN: BUILDING EMACS FROM SOURCE ON WINDOWS

Introduction

This report provides a lesson plan for building the Emacs text editor from source on a Windows machine, specifically version 30.1. The lesson plan is designed to be delivered asynchronously via OBS (Open Broadcasting Software) and is structured in the BOPPPS (Bridge-in, Objectives, Pre-assessment, Presentation, Practice, and Summary) format.

Please note that the plan in this report should be amended based on the circumstances of the build. For example, the choice between WSL (Windows Subsystem for Linux) or MSYS2 should be inline with the most convenient system architecture.

The Lesson Plan

Lesson Metadata

Course	Mastering Emacs for Creatives
Lesson	Building Emacs 30.1 from Source on Windows
Time	~1.5 hours, split into ~eight ten-minute videos
Resources Required	• Browser to watch the video. • Computer with Windows and WSL or MSYS2 installed. • `Git`, `Wget`, or Curl to retrieve the source files • A program like 7zip or Gzip to extract the files. • Any basic editor to document issues during the build (always recommended).
Resources Provided	• The video URL • Downloadable transcripts • A document written by the instructor outlining tips and tricks • Links to authorative articles on the topic, such as documentation for WSL, MSYS2, and Emacs 30.1 • A downloadable repository with the code at various before and after steps.

Lesson Bridge In

Instructor Activities	Time
Instructor Discusses Objectives	5 minutes

Picture this: You're a writer working on your next magnum opus, or a programmer tackling a complex coding project, but the deadline is approaching fast and you're not even halfway there! If only you could speed things up!

Enter Emacs. With Emacs on your system, you will be able to achieve 10x more in the same amount of time, or, if you prefer, the same amount of work in 1/10 th of the time.

Emacs is an incredibly flexibile, customizable text editor that's tailored to your specific needs. With Emacs, your workflows can be streamlined and your productivity can skyrocket.

By building Emacs on your local system, you'll gain access to its full range of powerful features and integrations, allowing you to move paragraphs and text around with ease, store and retrieve information instantly, and navigate your computer with lightning speed.

In today's lesson your going to learn to build Emacs from source on your Windows machine. Building from source allows you to customize Emacs to the exact specifications of your computer's architecture, ensuring it runs efficiently on your hardware.

Historically, it's easier to build Emacs on Linux, but Windows users can rely on WSL, MSYS2, or Cygwin to build emacs on windows, and get the benefits of a customized, more secure, and better performing build than they would get installing from a package manager.

In other lessons we cover how to install Emacs using package managers on both Linux and Windows. Mac users, while not the main focus today, should be able to follow along and implement the lesson with only some minor adaptations, particularly to ensure emacs plays nicely with the quartz compositor.

In this session, you are going to ensure the build environment, including the build tool chain on WSL or MSYS2, is up to date and in alignment, and then install the necessary packages and build Emacs 30.1 on Windows.

Lesson Objectives

Instructor Activities	Time
Instructor reviews learning objectives. Describes how the lesson will proceed.	3 minutes

- Participants will configure a build environment on MSYS2 or WSL
- Participants will install all necessary prerequisite libaries to successfully build the software.
- Participants will be able to download the source code for Emacs version 30.1 onto their Windows computer and check it is valid.
- Participants will follow the example provided by the instructor to build Emacs on their Windows machine.
- Participants will test that the build works.

Lesson Pre-Assessment

Given the lesson is intended for asynchronous learning sessions, the pre-assessment section will contain information about prerequisites, which will be presented to the audience in discussion format.

Instructor Activities	Time
Instructor discusses prerequisites. What the learners should know prior to attempting the lesson.	5 minutes

Build requires command of your system, which includes the read/write permissions to build Emacs in the MSYS2 directories, or in the Windows Subsystem for Linux (WSL) virtual machine managed by Hyper-V.

You should have enough space, processor speed, and memory on your computer to build Emacs. The instructor will demonstrate how to find the number of cores and check that the architecture satisfies requirements.

A little knowledge of Microsoft WSL and/or MSYS2 is necessary to get us to the point where we can begin the build and installation process.

While we will be discussing the WSL build, which is easier, in this session, we will use the MSYS2 build tool environment. Familiarity with GNU is recommended.

Before we begin, take a moment to write down for yourself what you want to take away from this lesson. Also open a text-editor to mark down the url, the time, the part you don't understand. What you did to resolve the issue (your prompt and/or the AI or website you used to resolve), and then take note of the outcomes. What worked, and what didn't work.

This will help you to learn to understand how these programs and systems work, and provide you searchable documentation you can use to revisit the lesson.

Lesson Participatory Learning

Instructor Activities	Learner Activities	Time
• Introduction to Windows Subsystem for Linux (WSL), MSYS2, and Cygwin - with focus on MSYS2 • Explanation of the GNU project • Demonstration of how to acquire and extract the necessary source files for the build	• Identify and implement methods for building Emacs on Windows using WSL/MSYS2 • Complete the compilation process step-by-step • Run various versions of Emacs (GUI/command-line) on their systems • Test functionality, fix errors, and optimize the build	~1 hour, 15 minutes

Lesson Participatory Learning Cont.

Instructor Activities	Learner Activities	Time
• Explanation of build tool chains, 'make', 'configure', 'clean', and 'install' • Explanation of gcc+ build tools, and build tool chain alignment • Warning about tool chains that are out of alignment • Demonstration of build tools installation • Explanation of configure, what it does • Depending on level of detail desired, explanation of each argument that can be used for configure (including custom optimizations for system) • Explanation of libraries that should be installed in advance, and what configure will do if there are present or not in an Emacs 30.1 build • Demonstration of building the software • Demonstration of testing the software • Demonstration of troubleshooting errors in a configure script (uninstall mandetory library and proceed with build, then reinstall)		

Lesson Participatory Learning Cont.

Instructor Activities	Learner Activities	Time
• Explanation of where the created files go and what to do with the source files. • Explanation of different types of executables (daemon vs executable) • Explanation of the key bindings problem that occurs when a program built to run on the GUI is run in command line mode • Rebuilding the program and configuring for optimal performance. • Explanation of installation process (copy on Windows, install on WSl, install on Linux) • Demonstration of launching the program to load a specific .init file		

Lesson Post-Assessment

Given the lesson is asynchronous, and the students will not be interacting directly with the instructor during the lesson, the post-assessment is an opportunity for the instructor to 'prompt' students in ways that might stimulate their learning.

Instructor Activities	Time
Instructor prompts students to reflect on learning outcomes	5 minutes

1. After completing this build, revisit your initial expectations. Did the process align with your assumptions? Did anything surprise you?

2. How does understanding the build process of Emacs relate to your broader understanding of software development and open-source projects?

3. Imagine you encounter an error during the build. Describe one potential error and how you might approach debugging it. Example answer: “I get an error about a missing library. I would check my package manager to confirm the library is installed. If it is, check my PATH variable, and check the version, possibly feeding the error message to AI. If it’s not installed, I would install it, then continue again.”

4. Why might someone choose to build Emacs from source instead of using a pre-built package?

Lesson Summary

Instructor Activities	Time
Instructor provides a summary of the lesson, emphasizing what the students 'could' have taken away	5 Minutes

By the end of this lesson, you could have gained a practical understanding of how to build Emacs 30.1 from source on a Windows machine utilizing either WSL or MSYS2. Did you?

Are you now capable of:

- **Identifying** and implementing methods for building Emacs on Windows using WSL/MSYS2?
- **Completing** the compilation process step-by-step?
- **Running** various versions of Emacs (GUI/command-line) on your systems?
- **Understanding** the benefits of building from source, such as customization and security?
- **Diagnosing** and resolving common build errors?

Media Examples

The lesson will be presented using a combination of live coding demonstrations and explanations, leveraging OBS (Open Broadcasting Software) to create a visually engaging video. The video will be edited with OpenShot editor and possibly DaVinci Resolve. Additionally, AI-powered video editors such as Capcut, Higgsfield, and/or Kokoro may be used to automate processes like background removal, subtitle design, and icon inclusion.<br>

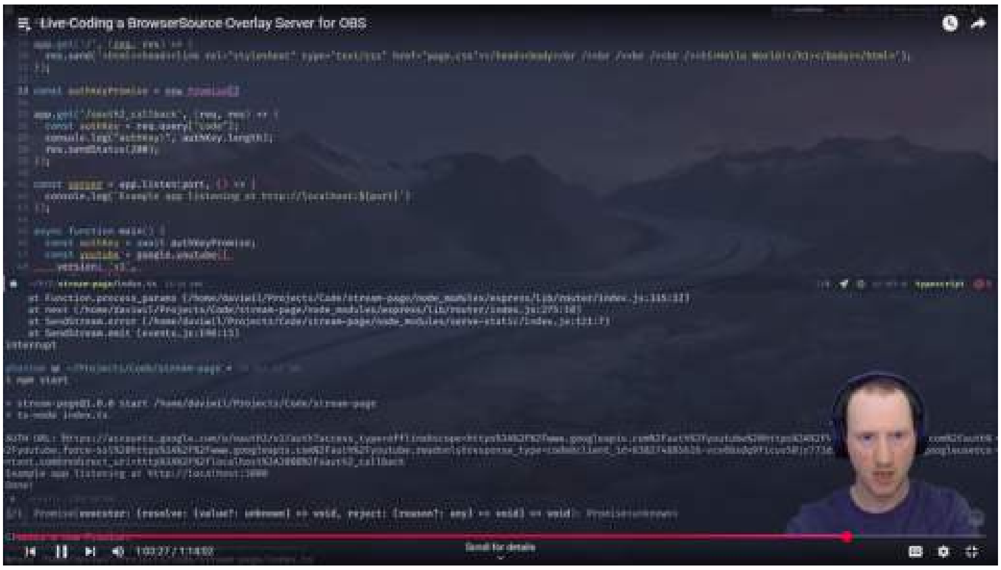
OBS Setup

The lesson will feature two primary views:

- **Full-screen speaker view**: The speaker will be displayed full-screen, with space on the side for text overlays. This will be used for explanations and discussions.
- **Overlay view**: The speaker will be overlaid on the bottom right of the screen, while the demonstration takes place in the Emacs text editor. This will allow students to see the code in action.

To achieve this, a green screen will be used in conjunction with OBS’s chroma keying feature. This will enable seamless transitions between the speaker and the Emacs editor.

Example of OBS Usage



```
app.get('/', (req, res) => {
  res.send('<html><head><link rel="stylesheet" type="text/css" href="page.css"></head><body><br /><br /><br /><h1>Hello World!</h1></body></html>');
});

const authKeyPromise = new Promise(

app.get('/oauth2_callback', (req, res) => {
  const authKey = req.query['code'];
  console.log("authKey!", authKey.length);
  res.sendStatus(200);
});

const server = app.listen(port, () => {
  console.log(`Example app listening at http://localhost:${port}`)
});

async function main() {
  const authKey = await authKeyPromise;
  const youtube = google.youtube({
    version: 'v3',
```

```
stream-page/index.ts
    at Function.process_params (/home/daviwil/Projects/Code/stream-page/node_modules/express/lib/router/index.js:335:12)
    at next (/home/daviwil/Projects/Code/stream-page/node_modules/express/lib/router/index.js:275:10)
    at SendStream.error (/home/daviwil/Projects/Code/stream-page/node_modules/serve-static/index.js:121:7)
    at SendStream.emit (events.js:198:13)
interrupt

$ npm start

> stream-page@1.0.0 start /home/daviwil/Projects/Code/stream-page
> ts-node index.ts

AUTH URL: https://accounts.google.com/o/oauth2/v2/auth?access_type=offline&scope=https%3A%2F%2Fwww.googleapis.com%2Fauth%2Fyoutube [illegible]
[illegible] redirect_uri=http%3A%2F%2Flocalhost%3A3000%2Foauth2_callback
Example app listening at http://localhost:3000
Done!

Promise(executor: (resolve: (value?: unknown) => void, reject: (reason?: any) => void) => void): Promise<unknown>
```

This YouTube video demonstrates what the final result might look like (with a different instructor) after using OBS with a green screen to create a professional-looking video.

Media and Resources Required

- OBS software for video recording and editing
- OpenShot editor and/or DaVinci Resolve for video editing
- AI-powered video editors (Capcut, Higgsfield, and/or Kokoro) for automation
- Emacs text editor for demonstrations
- Green screen and chroma keying equipment for recording

Conclusion

In this report, we provided a comprehensive lesson plan for building Emacs from source on a Windows machine. The plan is designed to be flexible and adaptable to different build circumstances.

We highlighted the benefits of building Emacs from source, including improved security, custom configuration options, and optimized performance.

The lesson plan is structured in the BOPPPS format, making it easy to follow and understand. By the end of this lesson, participants will have successfully built Emacs from source on their Windows machine and can then move on to the next lesson, in which they will begin learning to customize it to meet their specific needs.

References

Wilson, D. (2020, August 28). *Live coding a BrowserSource Overlay Server for OBS* [Video]. System Crafters. https://www.youtube.com/watch?v=rqwMpgTCOuM

AI Models Used In This Report: The ideas, structure, writing, and editing in this paper were performed by the author. Various AI models were used to research, collect, and verify data, format arguments, and grammatically structure content. Models used include: Qwen 2.5 235b a22b, Claude Sonnet 4 2025-05-14, Google Gemma 3 12b IT, Meta Llama 3.1 405b Instruct, Qwen 14B, Meta Llama 3.1 8B