Customizing JupyterLab using extensions

Alex Bozarth

Software Engineer
IBM Center for Open-Source
Data & AI Technologies
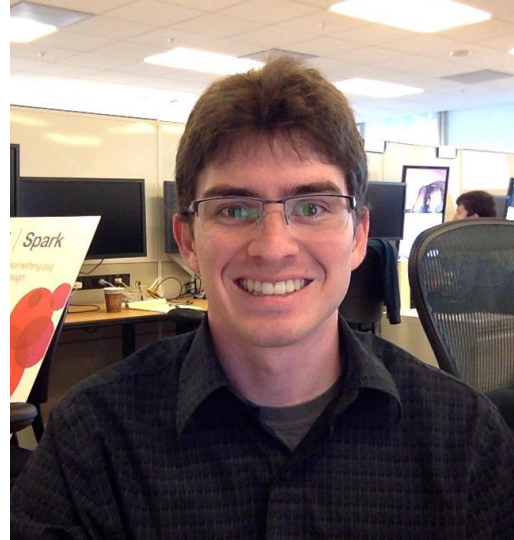
IBM **Developer**

IBM

About Me – Alex Bozarth

History

- Software Developer at IBM for 4 years
- PPMC for Apache Livy (Incubating)
- Open-Source Contributions to Apache Spark, Apache Livy, IBM Model Asset Exchange, IBM Data Asset Exchange, and JupyterLab

Current Work

- Creating AI-centric JupyterLab Extensions

Contact

- ajbozart@us.ibm.com
- @stbando
- github.com/ajbozarth
- linkedin.com/in/ajbozarth

Center for Open-Source Data & AI Technologies

CODAIT aims to make AI solutions
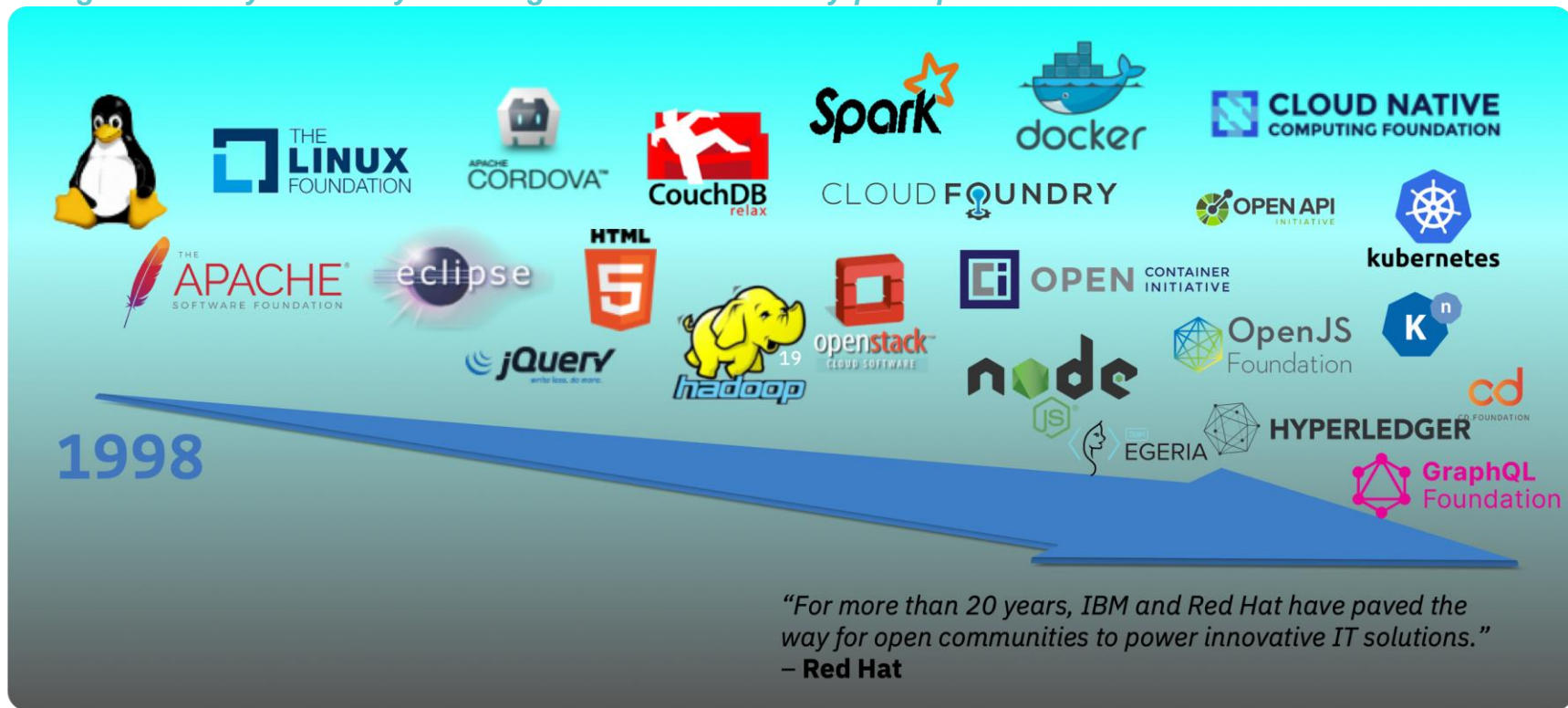dramatically easier to create, deploy,
and manage in the enterprise

–

Watson West Building
505 Howard St.
San Francisco, California

codait.org

IBM Open Source Participation

Long IBM history of actively fostering balanced community participation

"For more than 20 years, IBM and Red Hat have paved the way for open communities to power innovative IT solutions."
– **Red Hat**

Overview

The Questions

- What is JupyterLab?

- What are JupyterLab extensions?

- Why use JupyterLab extensions?

- When to create JupyterLab extensions?

The How To

- Getting started with JupyterLab

- Installing useful extensions

Going Further

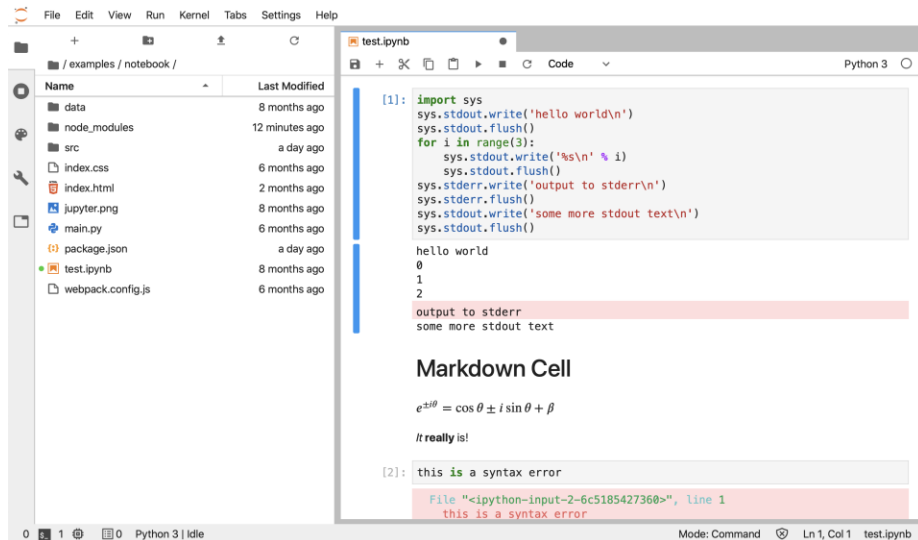- Creating an extension

Closing and Q & A

What is JupyterLab?

JupyterLab is the next generation UI for Project Jupyter

It provides a web-based interface for interacting with:

- Jupyter Notebooks

- Text Editors

- Terminals

- Etc.

It will eventually replace the classic Jupyter Notebook UI

JupyterLab extensions

What are JupyterLab extensions?

JupyterLab is designed to be extendable

Extensions enable users and developers to:

- Create new editors and output visualization
- Add buttons and menu items to existing functionality
- Provide APIs for other extensions to use

JupyterLab itself is just a collection of core extensions

Why use JupyterLab extensions?

The core extensions are intentionally limited in scope

Every user needs slightly different tools for their own work

By installing extensions specific to their scenario they can create a customized environment tailored to their work

Demo – Installing extensions

Let's show you:

1. Standard JupyterLab

   $ jupyter lab

2. Installing an extension via the UI – [TOC]

3. Installing an extension via CLI – [GIT]

   $ pip install jupyterlab-git <or> $ conda install -c conda-forge jupyterlab-git

   $ jupyter lab build

Demo — Creating an extension

Let's show you:

1. Creating a new extension using the cookie-cutter

   $ pip install cookiecutter

   $ cookiecutter https://github.com/jupyterlab/extension-cookiecutter-ts

2. Customizing your extension

3. Installing your extension (following the generated README)

4. Running JupyterLab with your extension

Q&A

Useful Links

JupyterLab Docs: https://jupyterlab.readthedocs.io/

Extension cookie-cutter: https://github.com/jupyterlab/extension-cookiecutter-ts

PyData Berlin Talk (about specific extensions): https://github.com/jtpio/a-tour-of-jupyterlab-extensions

GitHub Topics filter for finding extensions: https://github.com/topics/jupyterlab-extension

Gist with the demo button extension: https://gist.github.com/ajbozarth/48f0fe7f4213e11dc487046b73ded8f8

Link to download this ppt file: https://ibm.box.com/v/ajbozart-pydata19