Лекция 03

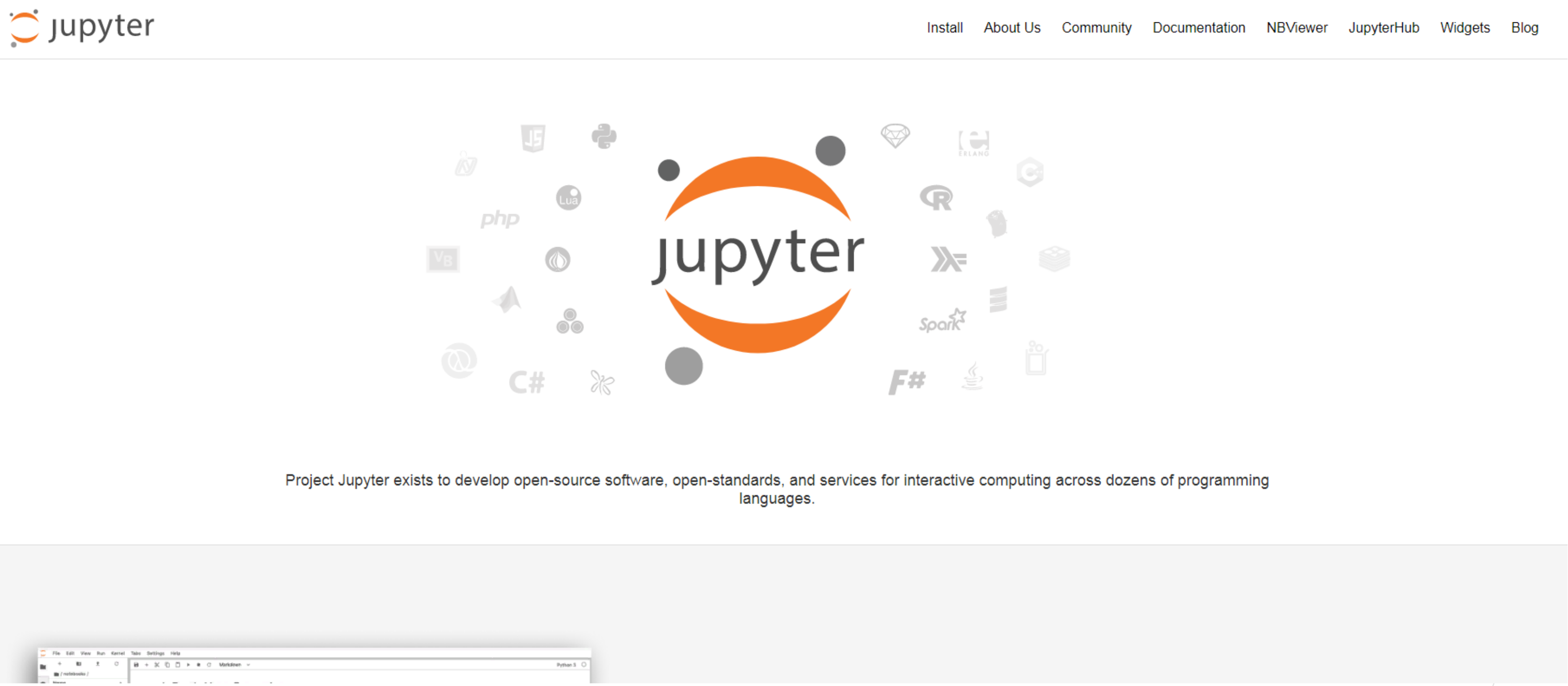
Библиотеки для анализа и обработки данных. MatPlotLib.

03

- Фрагмент 1 - Инструменты для написания кода и библиотеки
- Фрагмент 2 - Импорт и построение первого графика
- Фрагмент 3 - Базовые настройки отображения и использование MatPlotLib
- Фрагмент 4 - Отображение линий MatPlotLib
- Фрагмент 5 - Типы графиков MatPlotLib

#БиН

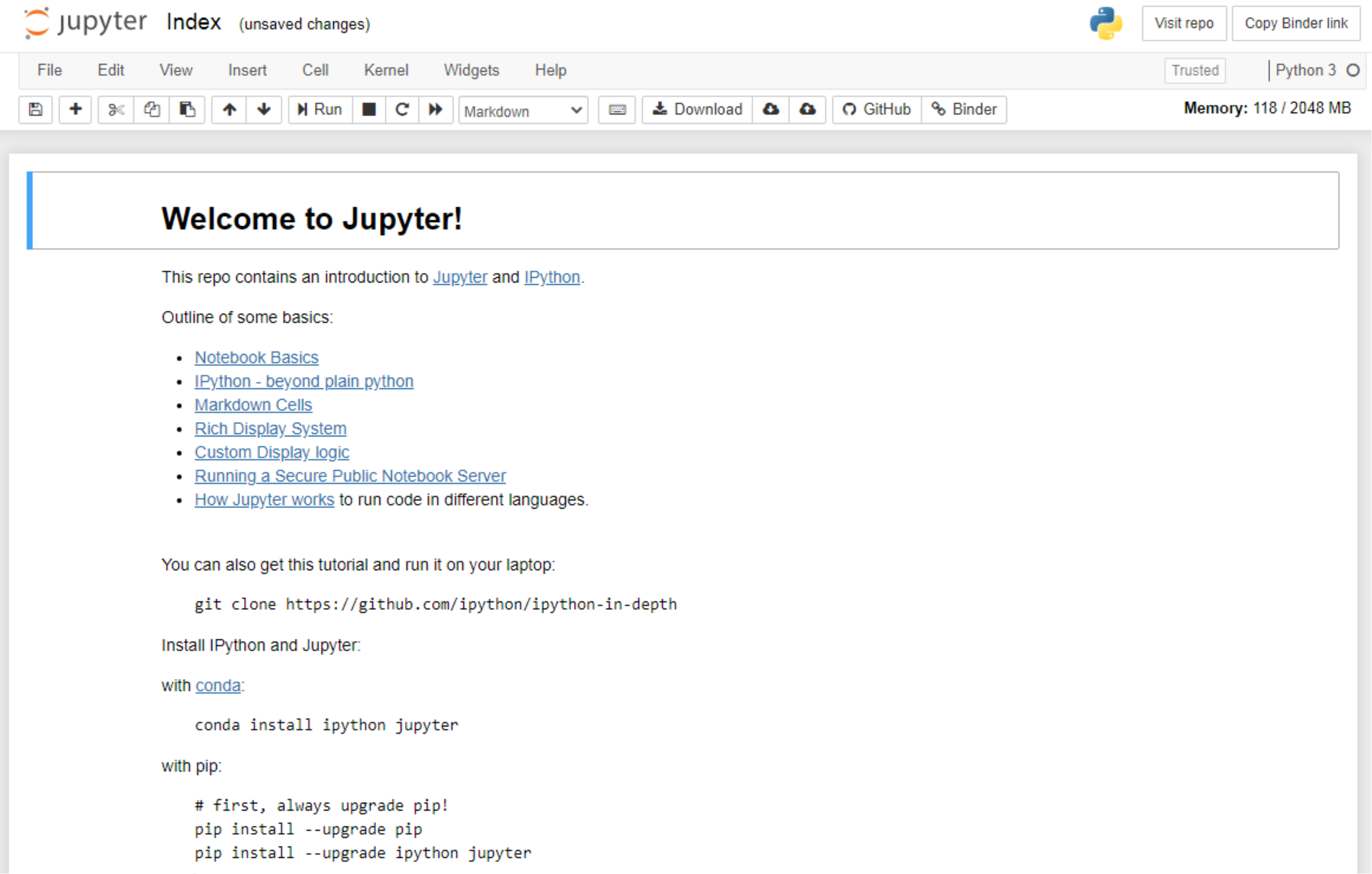
- https://jupyter.org/



Project Jupyter exists to develop open-source software, open-standards, and services for interactive computing across dozens of programming languages.

- pip install notebook
- pip install jupyter
- jupyter notebook

- https://jupyter.org/try



Welcome to Jupyter!

This repo contains an introduction to Jupyter and IPython.

Outline of some basics:

- Notebook Basics
- IPython - beyond plain python
- Markdown Cells
- Rich Display System
- Custom Display logic
- Running a Secure Public Notebook Server
- How Jupyter works to run code in different languages.

You can also get this tutorial and run it on your laptop:

```
git clone https://github.com/ipython/ipython-in-depth
```

Install IPython and Jupyter:

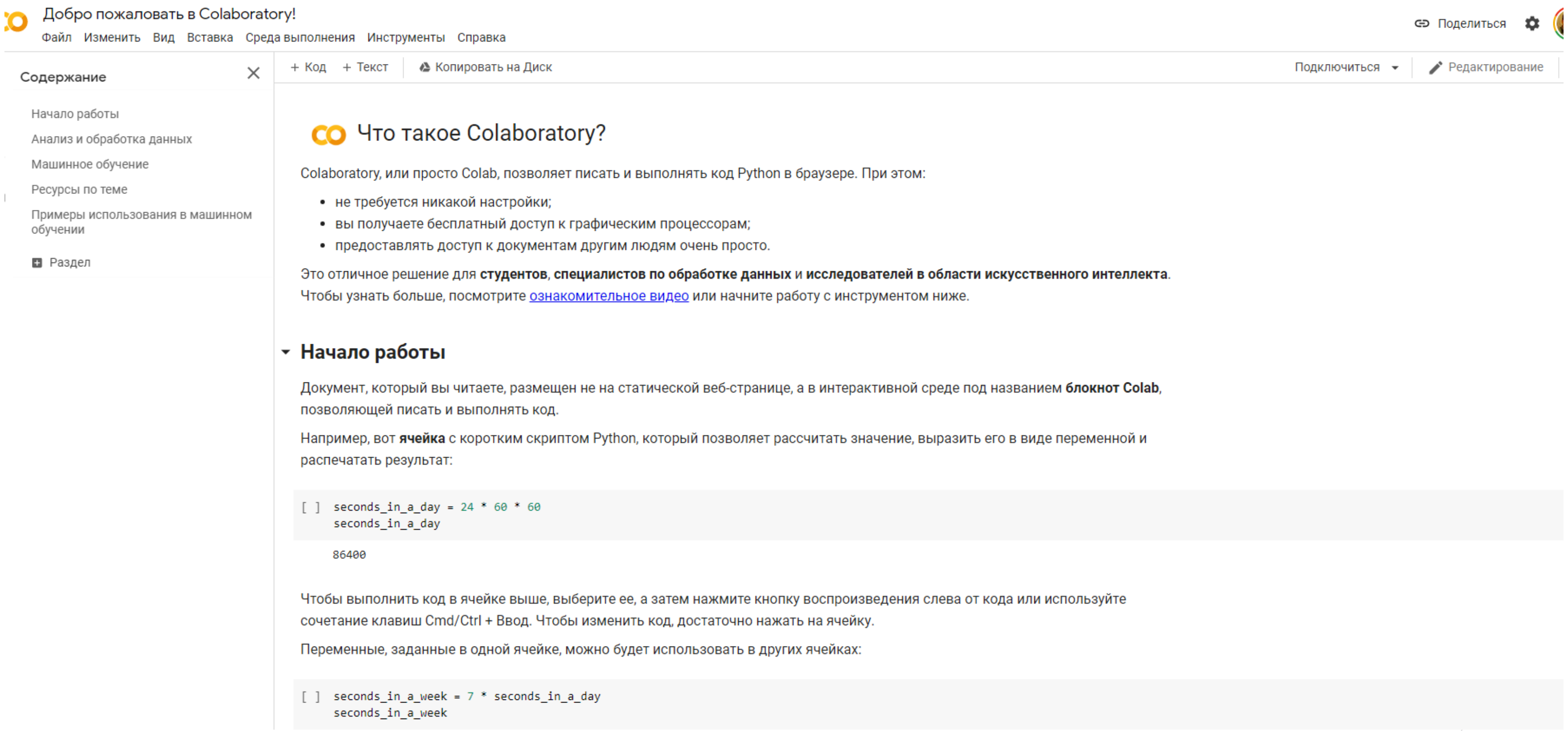
with conda:

```
conda install ipython jupyter
```

with pip:

```
# first, always upgrade pip!
pip install --upgrade pip
pip install --upgrade ipython jupyter
```

- https://colab.research.google.com/



Что такое Colaboratory?

Colaboratory, или просто Colab, позволяет писать и выполнять код Python в браузере. При этом:

- не требуется никакой настройки;
- вы получаете бесплатный доступ к графическим процессорам;
- предоставлять доступ к документам другим людям очень просто.

Это отличное решение для **студентов**, **специалистов по обработке данных** и **исследователей в области искусственного интеллекта**. Чтобы узнать больше, посмотрите ознакомительное видео или начните работу с инструментом ниже.

Начало работы

Документ, который вы читаете, размещен не на статической веб-странице, а в интерактивной среде под названием **блокнот Colab**, позволяющей писать и выполнять код.

Например, вот **ячейка** с коротким скриптом Python, который позволяет рассчитать значение, выразить его в виде переменной и распечатать результат:

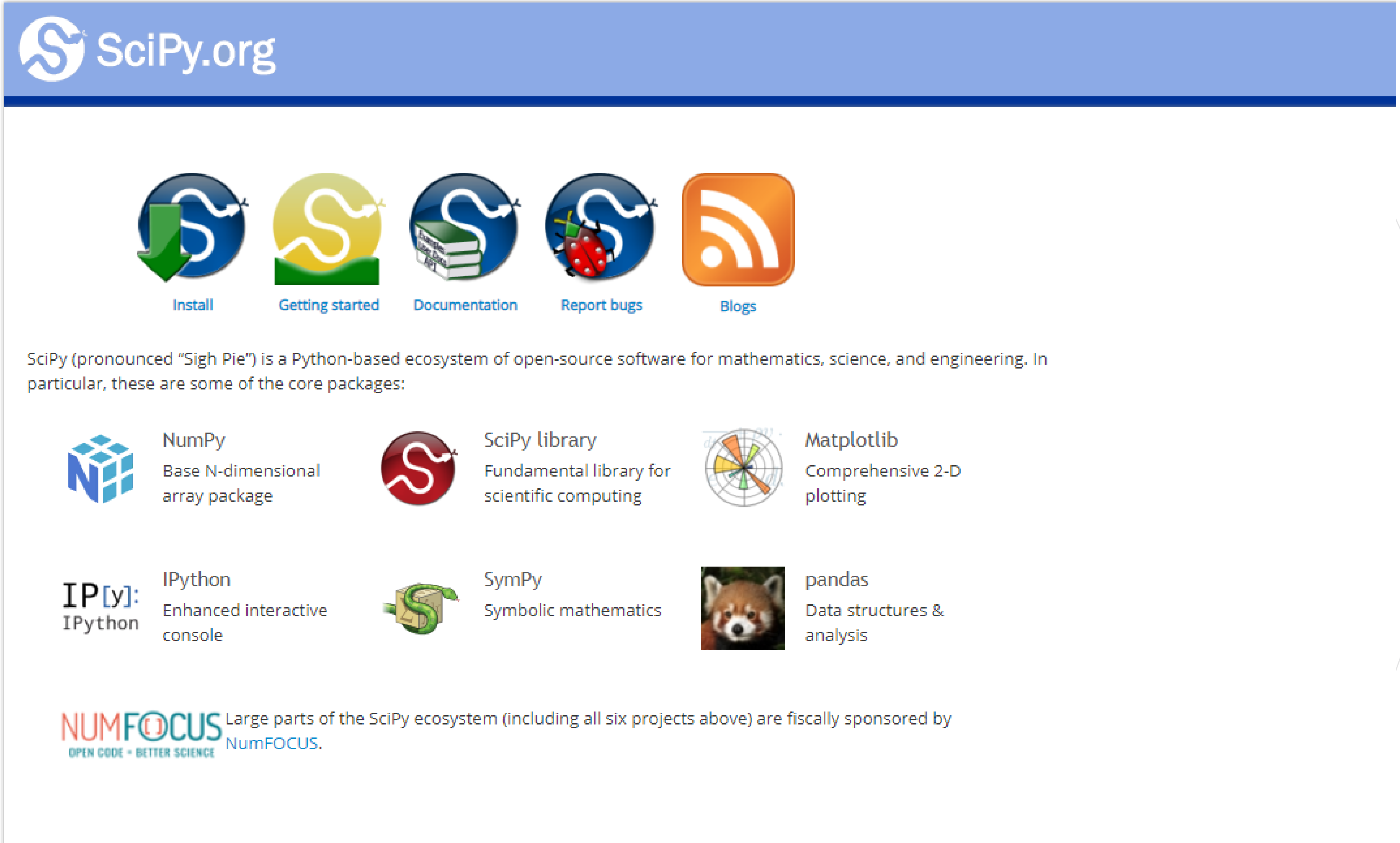
```
[ ]  seconds_in_a_day = 24 * 60 * 60
     seconds_in_a_day

     86400
```

Чтобы выполнить код в ячейке выше, выберите ее, а затем нажмите кнопку воспроизведения слева от кода или используйте сочетание клавиш Cmd/Ctrl + Ввод. Чтобы изменить код, достаточно нажать на ячейку.

Переменные, заданные в одной ячейке, можно будет использовать в других ячейках:

```
[ ]  seconds_in_a_week = 7 * seconds_in_a_day
     seconds_in_a_week
```

- https://www.scipy.org/



SciPy.org

Install
Getting started
Documentation
Report bugs
Blogs

SciPy (pronounced "Sigh Pie") is a Python-based ecosystem of open-source software for mathematics, science, and engineering. In particular, these are some of the core packages:

NumPy
Base N-dimensional array package

SciPy library
Fundamental library for scientific computing

Matplotlib
Comprehensive 2-D plotting

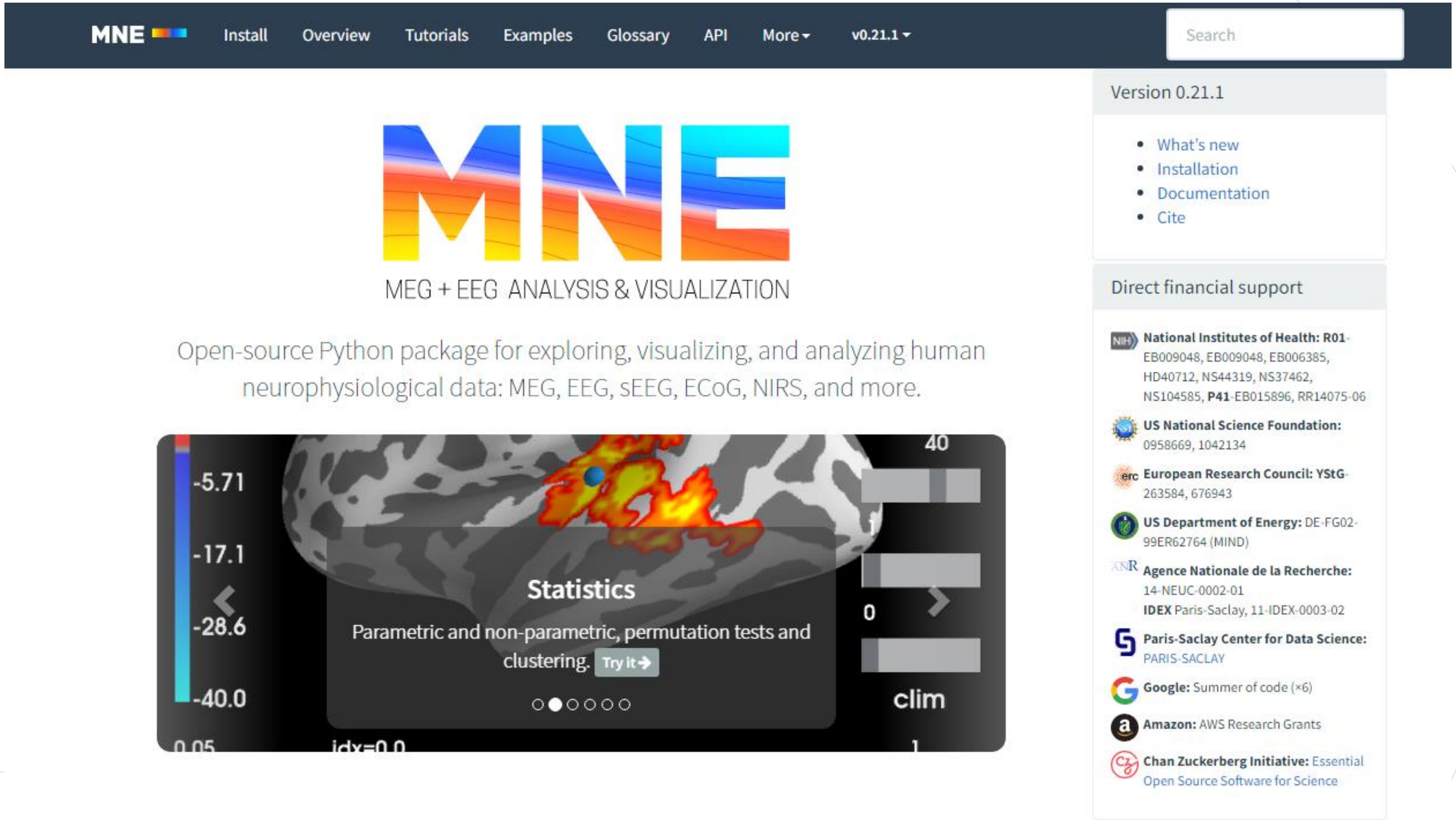
IPython
Enhanced interactive console

SymPy
Symbolic mathematics

pandas
Data structures & analysis

Large parts of the SciPy ecosystem (including all six projects above) are fiscally sponsored by NumFOCUS.

- https://mne.tools/stable/index.html



MEG + EEG ANALYSIS & VISUALIZATION

Open-source Python package for exploring, visualizing, and analyzing human neurophysiological data: MEG, EEG, sEEG, ECoG, NIRS, and more.

Statistics

Parametric and non-parametric, permutation tests and clustering.

Version 0.21.1

- What's new
- Installation
- Documentation
- Cite

Direct financial support

National Institutes of Health: R01-EB009048, EB009048, EB006385, HD40712, NS44319, NS37462, NS104585, **P41**-EB015896, RR14075-06

US National Science Foundation: 0958669, 1042134

European Research Council: YStG-263584, 676943

US Department of Energy: DE-FG02-99ER62764 (MIND)

Agence Nationale de la Recherche: 14-NEUC-0002-01
IDEX Paris-Saclay, 11-IDEX-0003-02

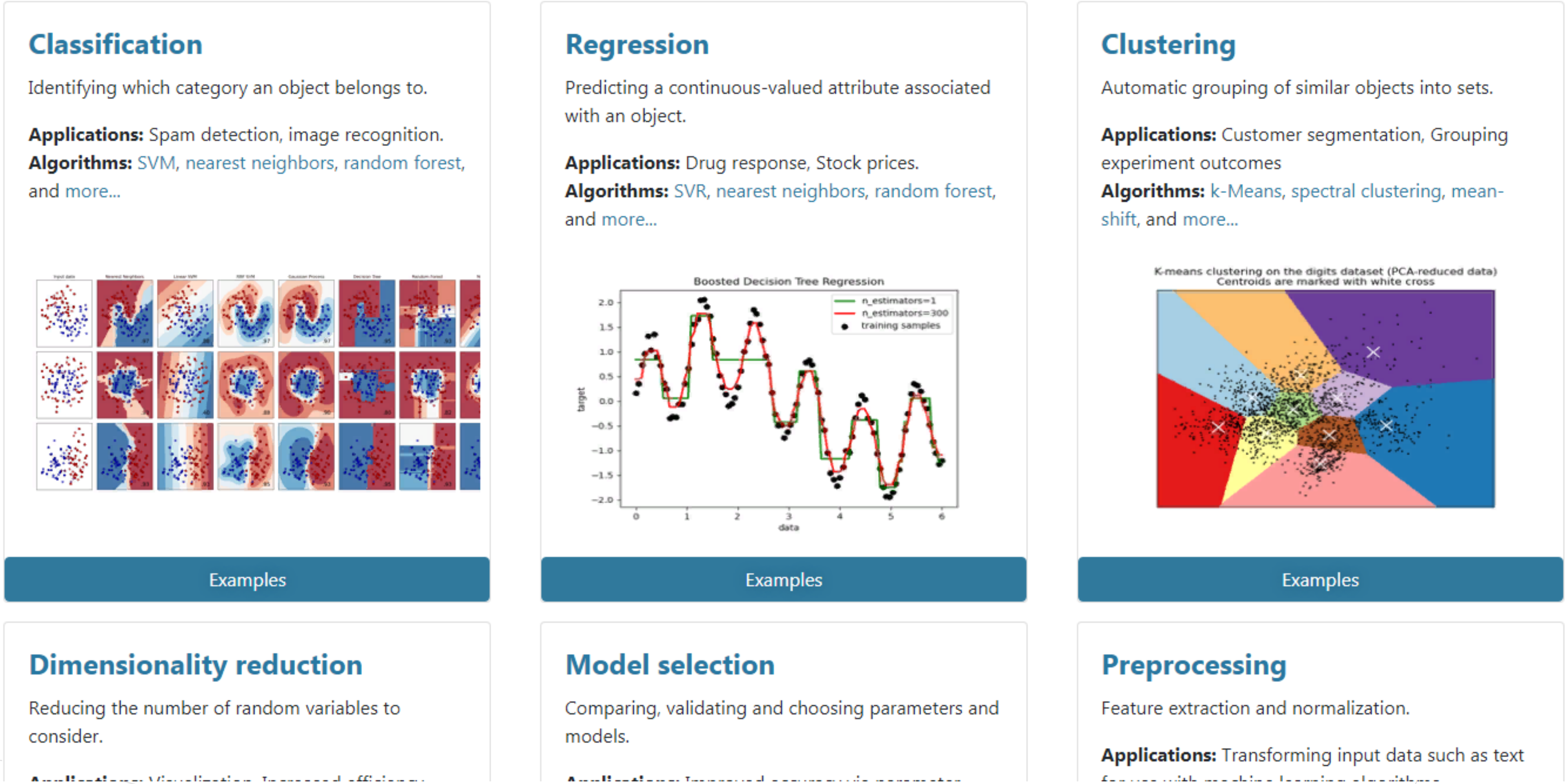
Paris-Saclay Center for Data Science: PARIS-SACLAY

Google: Summer of code (×6)

Amazon: AWS Research Grants

Chan Zuckerberg Initiative: Essential Open Source Software for Science

- https://scikit-learn.org/



Classification

Identifying which category an object belongs to.

Applications: Spam detection, image recognition.
Algorithms: SVM, nearest neighbors, random forest, and more...

Examples

Regression

Predicting a continuous-valued attribute associated with an object.

Applications: Drug response, Stock prices.
Algorithms: SVR, nearest neighbors, random forest, and more...

Boosted Decision Tree Regression

Examples

Clustering

Automatic grouping of similar objects into sets.

Applications: Customer segmentation, Grouping experiment outcomes
Algorithms: k-Means, spectral clustering, mean-shift, and more...

K-means clustering on the digits dataset (PCA-reduced data)
Centroids are marked with white cross

Examples

Dimensionality reduction

Reducing the number of random variables to consider.

Applications: Visualization, Increased efficiency

Model selection

Comparing, validating and choosing parameters and models.

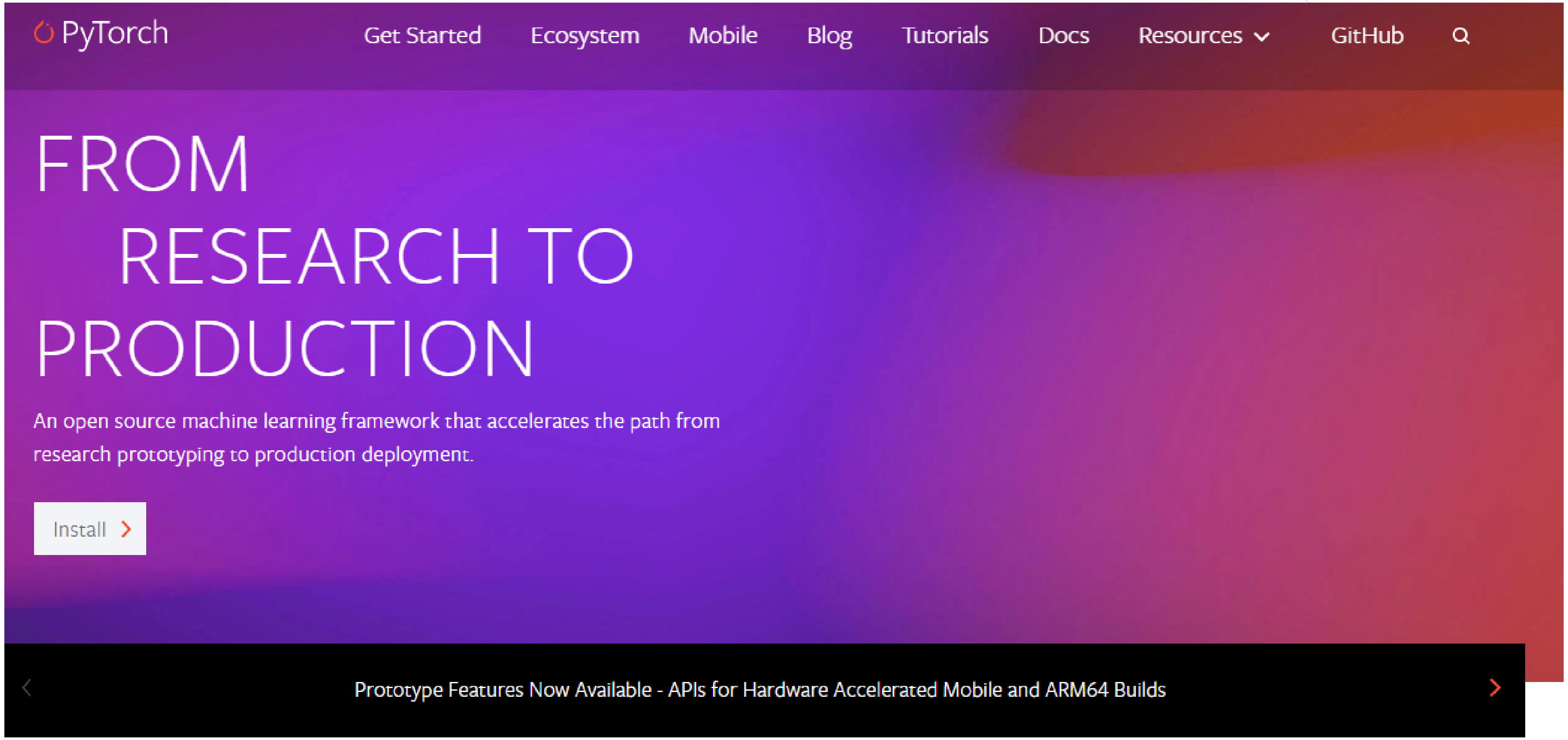
Applications: Improved accuracy via parameter

Preprocessing

Feature extraction and normalization.

Applications: Transforming input data such as text for use with machine learning algorithms

- https://pytorch.org/

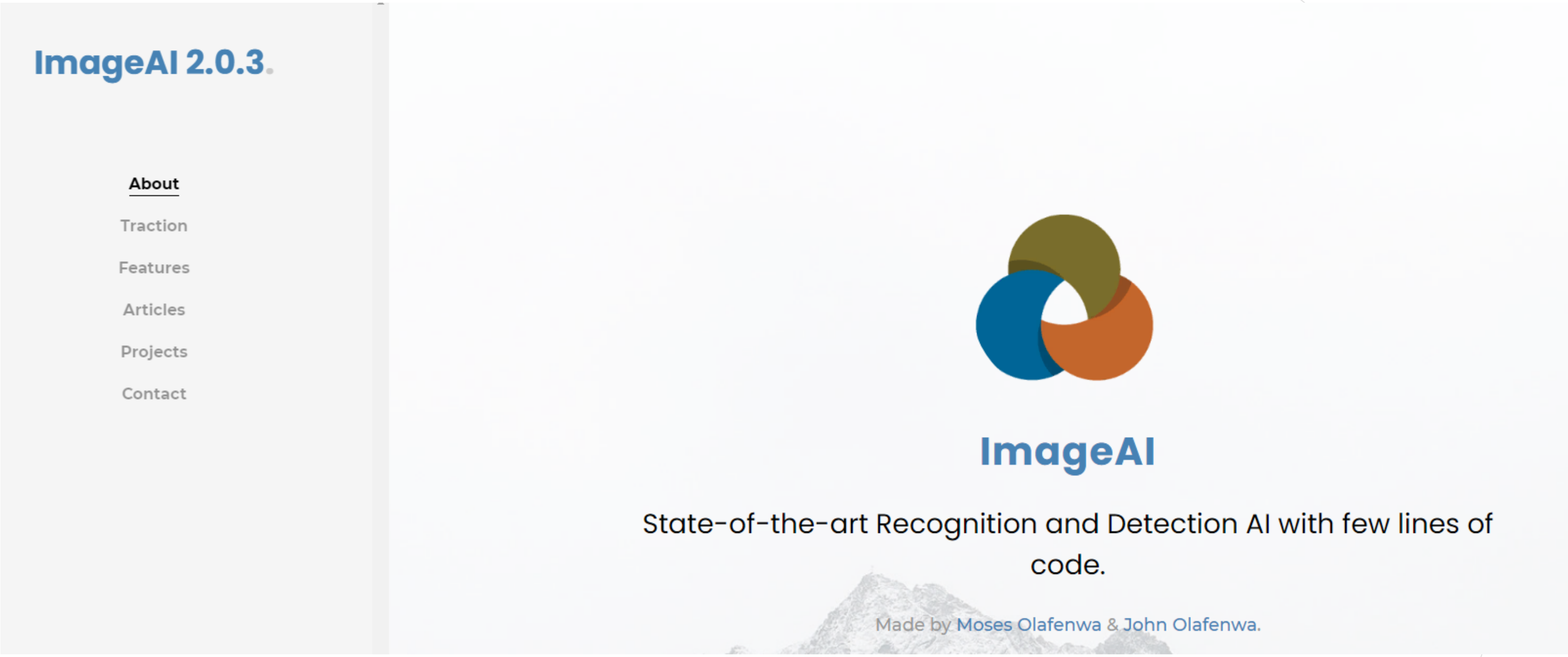


FROM RESEARCH TO PRODUCTION

An open source machine learning framework that accelerates the path from research prototyping to production deployment.

Prototype Features Now Available - APIs for Hardware Accelerated Mobile and ARM64 Builds

- http://imageai.org/



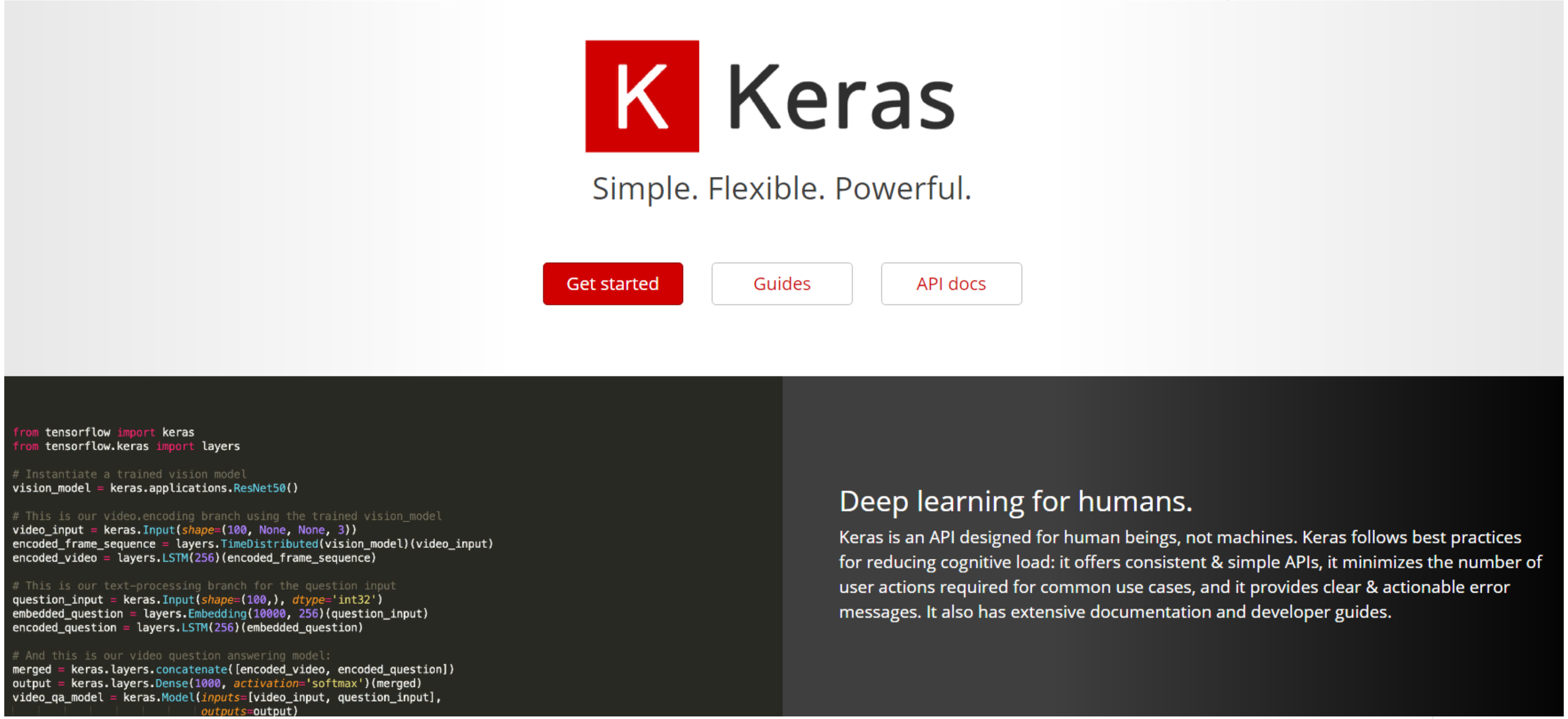
ImageAI 2.0.3.

ImageAI

State-of-the-art Recognition and Detection AI with few lines of code.

Made by Moses Olafenwa & John Olafenwa.

- https://keras.io/



Simple. Flexible. Powerful.

```
from tensorflow import keras
from tensorflow.keras import layers

# Instantiate a trained vision model
vision_model = keras.applications.ResNet50()

# This is our video.encoding branch using the trained vision_model
video_input = keras.Input(shape=(100, None, None, 3))
encoded_frame_sequence = layers.TimeDistributed(vision_model)(video_input)
encoded_video = layers.LSTM(256)(encoded_frame_sequence)

# This is our text-processing branch for the question input
question_input = keras.Input(shape=(100,), dtype='int32')
embedded_question = layers.Embedding(10000, 256)(question_input)
encoded_question = layers.LSTM(256)(embedded_question)

# And this is our video question answering model:
merged = keras.layers.concatenate([encoded_video, encoded_question])
output = keras.layers.Dense(1000, activation='softmax')(merged)
video_qa_model = keras.Model(inputs=[video_input, question_input],
                             outputs=output)
```

Deep learning for humans.

Keras is an API designed for human beings, not machines. Keras follows best practices for reducing cognitive load: it offers consistent & simple APIs, it minimizes the number of user actions required for common use cases, and it provides clear & actionable error messages. It also has extensive documentation and developer guides.