Docs as Part of the Product

Den Delimarsky, PM

@dennisc ode
Who am I?

- Program Manager at Microsoft
- Based in Vancouver
- Helping drive:
  - API Documentation
  - Samples
  - Interactive Experiences
State of Documentation
State of Documentation

• What we expect from doc experiences
  • Up-to-date and always reflecting the true state of the product.
  • Comprehensive.
  • Easy to edit.
  • Intuitive search and discovery.
  • Connected to the communities inside and outside the company.
  • Rich, interactive presentation.
State of Documentation

• **What we get** from doc experiences
  • Out-of-date – generated once and forgotten.
  • Inaccurate API docs written by hand.
  • Maintained in content silos.
  • Scattered – every team has their own site with their own format and publishing pipeline.
  • Search is bad due to fragmentation.
  • Text and basic media (images).
State of Documentation
Looking Back
Looking Back

- Started with the goal to be the one true place for all Microsoft developer resources.
- Powered by a closed, proprietary publishing system.
- Content stored in an internal XML flavor.
Looking Back

• Brittle code base not designed for the cloud.
• Everything is manually written – almost zero automation.
• Complicated process to update and publish content – sometimes it took days, if not weeks.
• Teams outgrew MSDN, held back by its update velocity – new sites started appearing.
Docs: The New Hope
• One doc site to rule them all – unify documentation for all company.
• Start from zero, for the cloud, from the cloud.
• Automate all the things.
• Open, using standard open-source tools and formats.
• Global by default – 64 locales built-in.
• We don’t know the right way – but we can experiment.
• Consistent editing experience – Markdown is the golden standard.
• Integrated in API reference (part of Javadoc comments, Swagger specs and Python docstrings).
• Edit directly in GitHub or favorite editor.
• Easily preview changes.
Getting started

You can optionally try the tutorials with pre-installed CNTK running in Azure Notebook hosted environment (for free) if you have not installed the toolkit in your own machine.

If you are coming from another deep learning toolkit you can start with an overview for advanced users.

If you have installed CNTK on your machine, after going through the installation steps, you can start using CNTK from Python right away (don’t forget to activate your Python environment if you did not install CNTK into your root environment):

```python
>>> import cntk
>>> cntk._version_
'2.5.1'
>>> cntk.minus([1, 2, 3], [4, 5, 6]).eval()
array([-3, -3, -3], dtype=float32)
```

The above makes use of the CNTK minus node with two array constants. Every operator has an `eval()` method that runs a forward pass for that node using its inputs, and returns the result. A slightly more interesting example that uses input variables (the more common case) is as follows:

```python
>>> import numpy as np
>>> x = cntk.input_variable(2)
```
Docs: The New Hope
• Automation at the heart of the publishing process
  • **API Doc Tooling** (Node, Java, Python, .NET, REST, PowerShell, CLI)
  • **Content Build and Validation**
  • **Content Testing Suite** (404s, orphaned pages, SEO compliance)
  • **GitHub Bots** (automatically merge PRs, channel external feedback to internal bug tracker)
  • **Sample Code Testing**
• Making URLs readable


https://docs.microsoft.com/dotnet/api/system.collections.generic.icomparer-1
• Convention over configuration – we infer content structure from folders in GitHub.
• `/content/test.md` becomes `docs.microsoft.com/cloud/content/test`
• Easy to set up redirects when things change, directly from the repo – broken links are much easier to fix.
Docs: The New Hope

• Content Versioning
  • No “burning in” into the URL.
  • Ensures URL consistency even when new versions are released.
  • Easily discoverable.
  • Reduces friction and broken links.
  • Using query param - ?view={moniker}
• API documentation discoverable from one place – the API Browser.
• No need to hop between N+1 sites to find the API.
• Semantic understanding of the APIs.
• Reduce discovery and documentation friction.
• Provide the artifacts (npm, pypi, source) and the docs are staged automatically.
• Intertwined with human-edited content.
Docs: The New Hope

- **npm Package or Source Code**
  - JSDoc/TypeDoc
  - Schema-based Document Processor
  - YAML + Markdown
  - docs.microsoft.com

- **PyPI Package, sdist/bdist or Source Code**
  - Sphinx
  - Schema-based Document Processor
  - YAML + Markdown
  - docs.microsoft.com

- **Maven Package or Source Code**
  - Javadoc/Doxygen
  - Schema-based Document Processor
  - YAML + Markdown
  - docs.microsoft.com

- **Swagger Spec**
  - Swagger Processor
  - Schema-based Document Processor
  - YAML + Markdown
  - docs.microsoft.com

- **NuGet Package or DLLs**
  - nu/doc
  - Schema-based Document Processor
  - YAML + Markdown
  - docs.microsoft.com
• **28K+** API documentation CI executed in the past year.

• **10MM+** lines of auto-generated docs dropped into GitHub.
Docs: The New Hope

• This powers:
  • 9.5K+ JavaScript API documentation pages
  • 55K+ Java API documentation pages
  • 16K+ Python API documentation pages
  • 15K+ REST API documentation pages
  • 499K+ .NET API documentation pages
• Builds run multiple times a day.
• Always documenting public latest versions of APIs in addition to secondary (supported) versions.
All API docs have standard URL patterns:

- /python/api/{package-name}/{entity}
- /java/api/{entity-qualified-name}
- /javascript/api/{package-name}/{entity}
- /rest/api/{product}/{op-group}/{operation}
- /cli/{product}/{command}
• Documentation linked to source code.
• Switch between versions on the fly.
• Logically grouped API entities in the table of contents.
• Grouping generated automatically – no human ever does that.
• Allows us to scale to 10K+ APIs in minutes.
• Contracts over hand-crafted documents.
• Schema defines entities and overall hierarchy.
• Template globally applied.
• Driving consistency in presentation.
• Updates don’t break existing documentation.
• Generate any post-processing artifacts after build – IntelliSense and cross-reference files.
• Artifacts can be used by product teams (Javadoc to be shipped with product).
• Structured documentation enables us to power rich API discovery experiences.
• Find the necessary API in seconds.
• Search across all products in a platform.
• IDE “auto-suggest” – in a search experience.
Beyond Text
Nation Shudders At Large Block Of Uninterrupted Text

WASHINGTON—Unable to rest their eyes on a colorful photograph or boldface heading that could be easily skimmed and forgotten about, Americans collectively recoiled Monday when confronted with a solid block of uninterrupted text.

(sourced from The Onion)
• Good documentation is not a wall of text.
• Reducing friction from reading to trying – how can we allow you to see how things work in seconds?
• Structured content allows us to understand where we can enable interactivity.
Beyond Text

- REST “Try It”
- Powered by Swagger specs.
- Run REST calls from a documentation page.
- Instantly see output, with no apps involved.
Beyond Text

- .NET REPL
- Run C# code in a stateless container.
- Zero friction to get started – no auth required.
- Any C# snippet can integrate it.
Beyond Text

- Azure Cloud Shell
- Linux in the browser.
- Works with Bash and PowerShell Core.
- Stateful container connected to Azure subscription.
Focus on Community
Focus on Community

- 2.5K+ repositories
  - 1.1K+ public
- 4.3K+ internal members
- A huge shift in how the entire company sees documentation and contributions to open source.
Focus on Community

• A lot of our projects were moved over to GitHub (VSCode, TypeScript, .NET, Monaco Editor).

• Natural place to have documentation, with a huge community of passionate developers.

(Stats courtesy of GitHub)
Focus on Community

- Shifting feedback from silo-ed platforms to be open.
- GitHub Issues – for content and site feedback.
- Documentation is treated like a product – doc issue = bug.
Focus on Community

• Key learning – transparency matters.
• Your customers know their needs better than you do – talk to them. All the time.
• Working with your community is not the same as asking them to do the work for you.
• Fostering the community and building trust takes time – coaching them on best practices and approaches is important.
Focus on Community

• Automation is your friend (again)
  • Contribution License Agreements (CLAs)
  • PR reviews ("Is my PR changing the right things?")
  • Content build validation ("Is what I added causing issues?")
  • Test any inserted code ("Does it build?")
<table>
<thead>
<tr>
<th>Feature</th>
<th>Before</th>
<th>After</th>
</tr>
</thead>
<tbody>
<tr>
<td>Open Source Docs</td>
<td>✗ No</td>
<td>✗ Yes</td>
</tr>
<tr>
<td>Localization</td>
<td>✗ Poor</td>
<td>✗ 64 Languages</td>
</tr>
<tr>
<td>Mobile Support</td>
<td>✗ None</td>
<td>✗ Major platforms</td>
</tr>
<tr>
<td>Accessibility</td>
<td>✗ Varied</td>
<td>✗ Built-in</td>
</tr>
<tr>
<td>Content Location</td>
<td>✗ Fragmented</td>
<td>✗ Unified</td>
</tr>
<tr>
<td>Sample Testing</td>
<td>✗ Sparse</td>
<td>✗ Automated CI</td>
</tr>
<tr>
<td>API Docs</td>
<td>✗ Manual</td>
<td>✗ Automatic</td>
</tr>
<tr>
<td>Feedback</td>
<td>✗ Varied, closed</td>
<td>✗ GitHub</td>
</tr>
<tr>
<td>Analytics</td>
<td>✗ Fragmented</td>
<td>✗ Unified</td>
</tr>
<tr>
<td>Engineering</td>
<td>✗ Duplicated</td>
<td>✗ Shared</td>
</tr>
</tbody>
</table>
Handling Legacy Resources
Handling Legacy Resources

Expectation

Reality
Handling Legacy Resources

• Mo’ sites, mo’ problems.
• Not as simple as shutting the old site down in favor of the new one.
• Content migration takes time – you will discover problems. A lot of problems.
• Redirection is important – customers don’t like broken links. Neither do search engines.
• Links are “baked into” products over years – you don’t want to break those.
Handling Legacy Resources

• You will inevitably get feedback that “old was better” – that’s not a cue to rebuild the old experience on the new site.
• Communication is important – set expectations.
• Habits die hard – it will take time for people to rely on new workflows.
Contact & Resources

@denniscoode
@docsmsft
https://docs.microsoft.com
https://aka.ms/docfx
dendeli@microsoft.com