Tom Johnson Q&A with Nytril

What problems does Nytril solve? 1
Example of multiple documents from data 1
What can Nytril do, that others cannot? 2
Starting out 2
Getting your data 3
Document Output 4
Localization 4
Workflow 5
AI advantage? 6
Pricing 6

1. What problems does Nytril solve?

> *TJ asks* - Who did you build Nytril for, and what problem were you trying to solve? What tools were you using before, and where did they fall short?

Our team has been involved with creating software, user manuals and training courses for decades. In the early days, our materials were created with Word and RoboHelp with some CI automation.

The core problem was the inability to separate and reuse data, formatting, language, and images. We wanted to reuse content across different versions and types of user-facing materials, but all the information was 'stuck' in our documents. Language translation was done at the document level, costly and a huge impediment to fixing problems that were discovered after the documents were translated.

We developed Nytril to solve these and other problems in a graceful, powerful and complete way. It is designed for technical writers who want to build and manage a knowledge-base and use it to create documents in many customized forms and languages. It is for those who want complete control over data, graphics, styles, and the whole document creation process.

2. Example of multiple documents from data

> *TJ asks* - You describe Nytril as "Excel plus Word" or "Markdown plus Python." Can you walk through a concrete example, such as generating 100 one-sheets and a product catalog from the same source? What does the source look like, and what happens when a product spec changes?

To illustrate how this works, this document was coded in Nytril. The questions and answers are shown together in this document, but there are 10 other documents that show each question by itself, to demonstrate content reuse. Please take a look at the source code, shown at the end of this document, to see how this looks.

If you take the code by itself, and run it in Nytril, you will see this document appear on screen. In the IDE menu, you can choose any of the 10 other documents to view. If any text or formatting is changed in the code, then all 11 documents are automatically updated. It is in that sense that Nytril is like in a spreadsheet. Change the data and/or code then update to instantly see the results.

We approached writing the Nytril code in this way:

1) Define a class that holds the contents of each item (i.e. Question and Answer).
2) Import the question text into Nytril with a 'Paste Special' feature that converts text from the clipboard into Nytril source.
3) Create the main document by writing a loop to iterate over and format each question.
4) Loop over each question again to create the 10 individual question documents.
5) Add a statement to reflect the source code itself, into the document.

A good real-world example of this, at scale, is the Nytril site for Sparling Instruments that contains dozens of spec sheets that are driven by custom data structures. Those sheets stand alone, but are also combined in a single document with a Table of Contents to make a product catalog. If any piece of data, styling or text changes, then the entire set of documents can be rebuilt and published locally and online in a few seconds.

3. What can Nytril do, that others cannot?

> *TJ asks* - Many tech writers already single-source content with variables, content references, and conditional text in tools like DITA, MadCap Flare, or static site generators. What can Nytril do that those tools can't? When would you tell someone to stay with what they have?

Nytril does not store structured data in XML or in a proprietary database and build a document from that source. Instead, a document is a program that executes, building content along the way. Everything that goes into the document is visible to the writer. A built-in debugger lets you trace each step as the code runs.

- There is no data-entry UI.
- Data and text are expressed in source code.
- All the files are on every writer's machine.
- Writers can edit offline and merge later.
- There is no cloud content management system, but Git can be used to track changes.
- There is full programming power at every content level from data to formatting to document export.

Nytril works the same way as a software project. You organize your data, text, variables, and definitions in source code and use source control. Image files, data in databases and web APIs can be easily accessed by code to add to your content. Since Nytril is a full programming language, calculations and logic can be freely mixed with content without "escaping" to another language or grammar. In the Mortgage Calculator program, for example, Nytril collects user input, calculates the numbers and formats the document: all in one small program.

Despite its power, Nytril is different from other approaches and, like any system, has a learning curve. People should stick with what they have if they do not want to incur switching costs or they do not need the extra power.

4. Starting out

> *TJ asks* - Nytril is a programming language with C#-style syntax. How much programming does a writer need to know? What does the first week look like for someone who has only written Markdown?

Just like Markdown, all content and formatting in Nytril are described in plain UTF8 text. The most obvious difference from Markdown is that literal text strings are in quotes. Variables, functions and operators are undecorated, and spaces and line breaks are insignificant.

Markup languages, such as Markdown, LaTeX, Typst and HTML, default to text and then require an 'escape sequence' for formatting, logic and everything else.

It is fair to say that a plain Markdown document seems simple compared with Nytril code containing the same text. However, once writers start inserting fields and formatting, adding external data, queries, images, and graphics, the Markdown approach loses all of its simplicity and charm. It is in the realm of complex data-driven documents that Nytril ends up with the simplest source expression.

Remarkably, there are no 'Write' or 'Print' statements in Nytril. If you want to put a string or the value of a variable into your document, just type the name, and its value flows into the body. If you want to add π to an angle, or convert temperature to Celsius, you don't need to escape to call python or JavaScript. You simply put the formula right in line with your text, and the computed value appears in your document.

Importantly, if you hold down the Ctrl-Key in the IDE and click anywhere in the document, Nytril will take you to the place in the source code that caused that content to be there. This feature allows a writer to 'click around' in an existing document to figure out where things are and how they work.

Reusing common content in Nytril is easy, define it in one place, and use the variable name in as many places as necessary. This not only works for text, but for numbers, functions, formulas, charts, paragraphs and even entire documents.

Content Creators can start using Nytril with no programming experience. The coding power is always there, but it is not always required. Starting out in a realistic project, writers would likely bootstrap with an example template, and modify it. Nytril has hundreds of example scripts and open-source templates to choose from, with more being created all the time.

5. Getting your data

> *TJ asks* - Where does the data come from? Can Nytril pull from a product database, spreadsheet, or API at build time, and how do writers and engineers share ownership of that data?

Nytril can query any local or online database to get a view. The writer supplies the SQL query and Nytril maps the view to defined structures. Nytril allows creators to connect to any Rest API or call any DotNet library. There are many open-source examples of this, shipped with the product. For security, there is a feature that allows each writer to securely cache their own access credentials on their workstation to avoid placing such credentials in source.

While Nytril has the ability to easily read and map XML and structured text files, we believe that creators should consider storing their data right in Nytril source, as we have done for this Q&A example. What happens next depends on the data, of course. Nytril provides formatters for tables and charts so you can reflect the data into a document. You can use normal programming constructs like loops, switches and logic to transform your data anyway you like without leaving Nytril code.

Here is an example of a research white-paper created with Nytril. The data for the paper was stored in Nytril source, exported to a program in C++ for heavy calculations, then automatically formatted to create the final paper. Nytril natively supports citations, references and bibliographies.

Here is an example of a IOT Spec Sheet created with Nytril. The data for the sheet is stored in nytril code and uses local image files. All the graphics, text and logos are defined in Nytril code.

6. Document Output

> *TJ asks* - What output formats does Nytril support? Can I publish HTML to my own site instead of nytril.com, and how does that HTML do on accessibility and mobile? If I stop using Nytril, how do I get my content out?

Nytril has a 'Publish' feature that creates all versions of a set of documents and stores them to a local drive and online, with one click. The user can choose PDF, DOCX or HTML as a local output format. Documents published online look great both on desktop and mobile devices.

Companies may wish to export locally and then use their own site for hosting. Nytril also supports a 'white-label link' to our site. This is a special link that writers put once on their company web site. It takes visitors to a 'Document Portal' with your company branding and colors. This feature allows the visitors to benefit from bookmarking, PDF download, localization and live document walk-through.

Nytril also has a quick export feature in the IDE, that exports the currently rendered document into various document formats, including image formats, LaTeX, SVG, Text, and more. This allows developers to test a single document in an exported format, before publishing all documents.

The simplest way to recover content from Nytril, if moving to another system, is to publish all documents to PDF or DOCX and import them to the new system. If custom data structures or language tables exist, it is easy to use Nytril to loop over the data and write it out to an external file or database. Since Nytril is new, we have had no customers do this yet, but exporting data into usable forms is very easy.

7. Localization

> *TJ asks* - How does Nytril handle localization, measurement units, and regional paper sizes? How does the integrated translation service work?

The most powerful Nytril documents are created in a way that supports text reuse and localization. In a localized project, user-facing language strings are stored together in a separate file. These are simply source files, but all the strings are given variable names in a `Lang` namespace. For example, instead of embedding the text — `"License Agreement"` — in your document, you would use a symbol such as `Lang.LicenseAgreement`.

A built-in feature of Nytril lets you easily gather up quoted strings from your project source files and make this language file for you. Another built-in feature will translate those strings to another language.

1) Choose 'Translate Language Strings' from the menu.
2) Choose the languages to translate into
3) Click 'Translate'
4) Review and edit the newly translated strings in source code

In the end, the project will have one source file for each supported language. Next, writers define Configuration objects in source code, to control which versions of each document they want to publish.

A simple example is a project with two configurations:

1) English, Imperial units, Letter paper
2) German, Metric units, A4 paper

Once defined, Nytril will automatically publish two different versions of each document. The correct version is shown to the customer based on their site preferences.

Nytril has full support for units and conversions between them. Most shops will want a customized way of dealing with the display of numbers with units, and this can all be easily handled with no limitations. In particular, the Sparling example above has custom routines for converting fractional inches to metric, in ways that handle industry specific edge cases. The important thing is that this code is written once, it is under your control, and can be used on all the project content, including external data.

8. Workflow

> *TJ asks* - How does Nytril fit into a docs-as-code workflow? Is the source plain text that diffs cleanly in Git, and can builds run headlessly in a CI pipeline?

Nytril source code files are plain UTF8 text. They seamlessly operate with Git and other systems. Here is the Nytril Template and Example repository on GitHub so you can see Nytril code in a repository.

Text review and approval is facilitated with the 'revu' string tag. As an example, this string is marked for review. It will show as purple in documents to make it easy for reviewers to identify. A ctrl-click on the text in the document will take the reviewer to the source string for editing.

The Nytril IDE runs natively on Windows, Mac and Linux desktop systems. As described above, it operates as its own document pipeline. We are looking forward to getting the first request for integration into a working CI system.

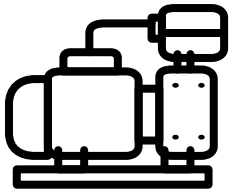
9. AI advantage?

> *TJ asks* - One possible advantage of Nytril being a programming language is that writers might not need to learn it in depth. They could describe what they want and let an AI coding tool write the Nytril code. AI tools are also strong at writing code, so documents defined in code might give them more to work with than prose-based formats. Is that a realistic workflow today, given that Nytril is new and LLMs haven't seen much of it in training? What are you doing to make Nytril easy for AI tools to work with (for example, reference docs, examples, or an llms.txt file)?

You are right about the potential. Given that Nytril allows you to specify the entire document in code, it allows an AI to mix queries, logic, translations and formatting together to create code for the requested documents.

So far, we have been able to get LLMs to understand the content of our site. Since our site is simply a big Nytril project, this means that users who publish to the site will benefit from having their content published in a way that is digestible to Google and AI crawlers.

As far as AI generated Nytril code, we have had success there as well. The following diagram was generated by ChatGPT after reading our tutorial document, and being asked to produce a drawing of a machine in Nytril code. The generated source for this diagram is at the very end of this document.



It is early days on the LLM front, but we foresee that AI systems will be able to deliver documents in 'source form' rather than as DOCX or PDF files. This is vastly more useful for content creators. It will allow writers to edit and correct the inevitable hallucinations, and use that work as a permanent resource for future documents.

10. Pricing

> *TJ asks* - Who is using Nytril now, how does pricing work between the Community and Professional editions, and how should a team think about adopting a tool from a small, new company?

The Community Edition can be downloaded off the site for free and used by anyone. It installs in less than 1 minute. Please try it out.

For corporate teams, we welcome the use of the Community Edition for evaluation. The subscription is $9 per month to enable the Pro Edition. With a Pro Edition, teams will have access to translation and publishing/hosting services.

Nytril is newly released, but has been in development and beta testing for years. It was very important to us that the product was polished before we started to spread the word. We do not have a lot of users yet, but the ones that we do have are enthusiastic, and provide great feedback. Nytril LLC is a small company, and therefore reacts rapidly to user requests and bug reports. Giving writers power to create is our mission.

Becoming an early adopter of new tech is an important decision for a team. It comes down to whether the increased productivity and/or lower cost of the new product is a greater benefit than the switching cost over some reasonable timeframe. I believe that once writers try it out, and understand the productivity and flexibility gains from this docs-as-code model, they will make the switch with enthusiasm and wonder how they got along without it.

```
using Format;

//=============================================================================
// Define all the documents in the project
//=============================================================================

readonly namespace Main.Documents {
  Questionaire MainDocument = new;
  QADocument[] QADocs       = new QADocument(each Interview); // 10 documents
}

//=============================================================================
// Define a structure to hold the Questions and Answers
//=============================================================================

class QuestionAnswer {
  var Guid Id;
  var      Name,
           Question,
           Answer;

  Constructor(Guid id, name) {
    Name = name;
    Id   = id;
  }

  // Code to format the Q&A
  Show(index) = Block {
    var inset = 24pt;

    Heading1 {
      DocFields.Anchor(Id);
      index;
      ".";
      LeftIndent: inset;
      FirstIndent: -inset;
      Tab;
      Name;
    };
    Paragraph(BorderLR(new(1, 8pt, #BBD4FF#))) {
      ParBackground: #EEF5FF#;
      ParAlignment: Justify;
      SpaceAfter: 8pt;
      LeftIndent: 48pt;
      RightIndent: 48pt;
      TextFace: Serif;
      Italic "TJ asks - ";
      Question;
    };
    Answer;
  };
}

//=============================================================================
// Formatting functions used in the answers
//=============================================================================

readonly Par = Paragraph {
  TextHeight: 12pt;
  LeftIndent: 8pt;
  SpaceAfter: 8pt;
  ParAlignment: Justify;
};

readonly Heading1 = Paragraph(BorderB(new(2, 2, 70%))) {
  KeepWithNext: true;
  TextFace: Serif;
  TextHeight: 16pt;
  SpaceAfter: 5pt;
  SpaceBefore: 16pt;
  Bold;
};
```

```
readonly NumberList = ListBlock(@Enumerators.NumberParen);

HyperLink(name, url) = Span {
  TextColor: Colors.Blue;
  name;
  TipAction: new URL(url);
};

DocLink(name, id) = HyperLink(name, "www.nytril.com/{0}"(id));

SCode(code) = Span {
  TextFace: Mono;
  TextColor: #74531F#;
  code;
};

CString(code) = Span {
  TextFace: Mono;
  TextColor: #A81535#;
  code;
};

//============================================================================
// The main document definition
//============================================================================

class Questionaire: DocumentEntry {
  Constructor {
    super.Constructor(#c3f708ac-388d-4d12-bcd3-54059d07d481#, "Q&A");
    Title = "Tom Johnson Q&A with Nytril";
  }

  override GetDocument = Document {
    // 3 sections, defined below
    TitlePage;
    QASection;
    SourceCodeSection;
  };

  SourceCodeSection = Block {
    SectionNextPage(24pt) {
      Header: MarginBlock {
        Paragraph {
          TextHeight: 9pt;
          ParAlignment: Right;
          "Nytril Source for this Q&A document";
        };
      };
      Footer: MarginBlock;
    };

    TextHeight: 8pt;
    SourceSelectionLines;
  };

  QASection = Block {
    SectionNextPage {
      PageNumberStart: 1;
      Header: MarginBlock {
        Paragraph {
          ParAlignment: Center;
          Title;
        };
      };
      Footer: MarginBlock {
        Paragraph {
          ParAlignment: Center;
          "- "; DocFields.PageNumber; " -";
        };
      };
    };
```

```
    foreach (var qa in Interview)
      qa.Show(each1);
  };

  TitlePage = Block {
    Paragraph {
      SpaceBefore: 2";
      SpaceAfter: 1";
      TextHeight: 32pt;
      ParAlignment: Center;
      Bold;
      Title;
    };

    // Table of contents
    Block {
      TextHeight: 16pt;
      foreach (var qa in Interview) {
        Paragraph(null, [new TabStop(6.5", Right, Dot)]) {
          SpaceAfter: 2pt;
          Span {
            Action: new Action(Actions.ToAnchor, qa.Id);
            qa.Name;
          };
          Tab;
          DocFields.PageReference(qa.Id);
        }
      }
    };
  };
}


//============================================================================
// Document that shows a single Q&A using different styling
//============================================================================

class QADocument: DocumentEntry {
  var QuestionAnswer QA;

  Constructor(QuestionAnswer qa) {
    super.Constructor(qa.Id, qa.Name);
    QA = qa;
  }

  override GetDocument = Document {
    Paragraph {
      ParAlignment: Center;
      TextHeight: 24pt;
      Bold;
      Name;
    };
    Paragraph {
      TextHeight: 20;
      SpaceBefore: 20pt;
      SpaceAfter: 20pt;
      TextFace: Serif;
      LeftIndent: 24pt;
      RightIndent: 24pt;
      QA.Question;
    };
    Block {
      TextHeight: 16;
      QA.Answer;
    };
  };
}
```

```
//========================================================================
// The 'raw data' for each Q&A
// Each object is given a GUID to identify it in the doc and online
// How these are laid out is a matter taste - spaces and lines do not matter
// 1 string per sentence faciliates translation and editing but is not required
//========================================================================

readonly QuestionAnswer[] Interview = [
  new(#597fcc39-cea3-490e-bb0d-6b22076157db#, "What problems does Nytril solve?") {
    Question: Span {
      "Who did you build Nytril for, and what problem were you trying to solve? ";
      "What tools were you using before, and where did they fall short? ";
    };
    Answer: Block {
      Par {
        "Our team has been involved with creating software, user manuals and training courses for decades. ";
        "In the early days, our materials were created with Word and RoboHelp with some CI automation. ";
      };
      Par {
        "The core problem was the inability to separate and reuse data, formatting, language, and images. ";
        "We wanted to reuse content across different versions and types of user-facing materials, but all the information was
'stuck' in our documents. ";
        "Language translation was done at the document level, costly and a huge impediment to fixing problems that were
discovered after the documents were translated. ";
      };
      Par {
        "We developed Nytril to solve these and other problems in a graceful, powerful and complete way. ";
        "It is designed for technical writers who want to build and manage a knowledge-base and use it to create documents in
many customized forms and languages. ";
        "It is for those who want complete control over data, graphics, styles, and the whole document creation process. ";
      };
    };
  },

  new(#ca7fde94-b8e7-44d1-9341-5f98f6f8f09f#, "Example of multiple documents from data") {
    Question: Span {
      "You describe Nytril as \"Excel plus Word\" or \"Markdown plus Python.\" ";
      "Can you walk through a concrete example, such as generating 100 one-sheets and a product catalog from the same source? ";
      "What does the source look like, and what happens when a product spec changes? ";
    };
    Answer: Block {
      Par {
        "To illustrate how this works, this document was coded in Nytril. ";
        "The questions and answers are shown together in this document, but there are 10 other documents that show each question
by itself, to demonstrate content reuse. ";
        "Please take a look at the source code, shown at the end of this document, to see how this looks. ";
      };
      Par {
        "If you take the code by itself, and run it in Nytril, you will see this document appear on screen. ";
        "In the IDE menu, you can choose any of the 10 other documents to view. ";
        "If any text or formatting is changed in the code, then all 11 documents are automatically updated. ";
        "It is in that sense that Nytril is like in a spreadsheet. ";
        "Change the data and/or code then update to instantly see the results. ";
      };
      Par {
        "We approached writing the Nytril code in this way:";
      };
      NumberList {
        "Define a class that holds the contents of each item (i.e. Question and Answer). ";
        "Import the question text into Nytril with a 'Paste Special' feature that converts text from the clipboard into Nytril
source. ";
        "Create the main document by writing a loop to iterate over and format each question. ";
        "Loop over each question again to create the 10 individual question documents. ";
        "Add a statement to reflect the source code itself, into the document. ";
      };
      Empty;
      Par {
        "A good real-world example of this, at scale, is the Nytril site for ";
          DocLink("Sparling Instruments", #67eadc1b-1214-4f27-a051-de966db75067#);
          " that contains dozens of spec sheets that are driven by custom data structures. ";
        "Those sheets stand alone, but are also combined in a single document with a Table of Contents to make a product
catalog. ";
```

```
        "If any piece of data, styling or text changes, then the entire set of documents can be rebuilt and published locally
and online in a few seconds. ";
      };
    };
  },

  new(#37787112-a604-4a53-a127-aed4d0b97b21#, "What can Nytril do, that others cannot?") {
    Question: Span {
      "Many tech writers already single-source content with variables, content references, and conditional text in tools like
DITA, MadCap Flare, or static site generators. ";
      "What can Nytril do that those tools can't? ";
      "When would you tell someone to stay with what they have?  ";
    };
    Answer: Block {
      Par {
        "Nytril does not store structured data in XML or in a proprietary database and build a document from that source. ";
        "Instead, a document is a program that executes, building content along the way. ";
        "Everything that goes into the document is visible to the writer. ";
        "A built-in debugger lets you trace each step as the code runs. ";
      };
      ListBlock {
        "There is no data-entry UI. ";
        "Data and text are expressed in source code. ";
        "All the files are on every writer's machine. ";
        "Writers can edit offline and merge later. ";
        "There is no cloud content management system, but Git can be used to track changes.";
        "There is full programming power at every content level from data to formatting to document export. ";
      };
      Empty;
      Par {
        "Nytril works the same way as a software project. You organize your data, text, variables, and definitions in source
code and use source control. ";
        "Image files, data in databases and web APIs can be easily accessed by code to add to your content. ";
        "Since Nytril is a full programming language, calculations and logic can be freely mixed with content without
\"escaping\" to another language or grammar. ";
        "In the "; DocLink("Mortgage Calculator", #ac4daea1-bb04-4b9d-9e48-62fe65f01f1f#); " program, for example, Nytril
collects user input, calculates the numbers and formats the document: all in one small program. ";
      };
      Par {
        "Despite its power, Nytril is different from other approaches and, like any system, has a learning curve. ";
        "People should stick with what they have if they do not want to incur switching costs or they do not need the extra
power. ";
      };
    };
  },

  new(#0c1e64ac-1fa4-4db7-909c-23f61bfba3e4#, "Starting out") {
    Question: Span {
      "Nytril is a programming language with C#-style syntax. ";
      "How much programming does a writer need to know? ";
      "What does the first week look like for someone who has only written Markdown? ";
    };
    Answer: Block {
      Par {
        "Just like Markdown, all content and formatting in Nytril are described in plain UTF8 text. ";
        "The most obvious difference from Markdown is that literal text strings are in quotes. ";
        "Variables, functions and operators are undecorated, and spaces and line breaks are insignificant. ";
        "Markup languages, such as Markdown, LaTeX, Typst and HTML, default to text and then require an 'escape sequence' for
formatting, logic and everything else. ";
      };
      Par {
        "It is fair to say that a plain Markdown document seems simple compared with Nytril code containing the same text. ";
        "However, once writers start inserting fields and formatting, adding external data, queries, images, and graphics, the
Markdown approach loses all of its simplicity and charm. ";
        "It is in the realm of complex data-driven documents that Nytril ends up with the simplest source expression. ";
      };
      Par {
        "Remarkably, there are no 'Write' or 'Print' statements in Nytril. ";
        "If you want to put a string or the value of a variable into your document, just type the name, and its value flows into
the body. ";
        "If you want to add π to an angle, or convert temperature to Celsius, you don't need to escape to call python or
JavaScript. ";
```

```
        "You simply put the formula right in line with your text, and the computed value appears in your document. ";
      };
      Par {
        "Importantly, if you hold down the Ctrl-Key in the IDE and click anywhere in the document, Nytril will take you to the
place in the source code that caused that content to be there. ";
        "This feature allows a writer to 'click around' in an existing document to figure out where things are and how they
work. ";
      };
      Par {
        "Reusing common content in Nytril is easy, define it in one place, and use the variable name in as many places as
necessary. ";
        "This not only works for text, but for numbers, functions, formulas, charts, paragraphs and even entire documents. ";
      };

      Par {
        "Content Creators can start using Nytril with no programming experience. ";
        "The coding power is always there, but it is not always required. ";
        "Starting out in a realistic project, writers would likely bootstrap with an example template, and modify it. ";
        "Nytril has hundreds of example scripts and open-source templates to choose from, with more being created all the time.
";
      };
    };
  },

  new(#8bd50f93-1087-4348-925a-07cc4aabc9e2#, "Getting your data") {
    Question: Span {
      "Where does the data come from? ";
      "Can Nytril pull from a product database, spreadsheet, or API at build time, and how do writers and engineers share
ownership of that data?  ";
    };
    Answer: Block {
      Par {
        "Nytril can query any local or online database to get a view. ";
        "The writer supplies the SQL query and Nytril maps the view to defined structures. ";
        "Nytril allows creators to connect to any Rest API or call any DotNet library. ";
        "There are many open-source examples of this, shipped with the product. ";
        "For security, there is a feature that allows each writer to securely cache their own access credentials on their
workstation to avoid placing such credentials in source. ";
      };
      Par {
        "While Nytril has the ability to easily read and map XML and structured text files, we believe that creators should
consider storing their data right in Nytril source, as we have done for this Q&A example. ";
        "What happens next depends on the data, of course. ";
        "Nytril provides formatters for tables and charts so you can reflect the data into a document. ";
        "You can use normal programming constructs like loops, switches and logic to transform your data anyway you like without
leaving Nytril code. ";
      };

      Par {
        "Here is an example of a "; DocLink("research white-paper", #d8c22a4e-4dde-44a7-b762-9a15a2de11fb#); " created with
Nytril. ";
        "The data for the paper was stored in Nytril source, exported to a program in C++ for heavy calculations, then
automatically formatted to create the final paper. ";
        "Nytril natively supports citations, references and bibliographies. ";
      };

      Par {
        "Here is an example of a "; DocLink("IOT Spec Sheet", #ee220d71-23c0-47fc-a7ef-e7f16244a6cf#); " created with Nytril. ";
        "The data for the sheet is stored in nytril code and uses local image files. ";
        "All the graphics, text and logos are defined in Nytril code. ";
      };
    };
  },

  new(#2d59c1cd-ac01-49d4-af4a-4ae20c25fbe6#, "Document Output") {
    Question: Span {
      "What output formats does Nytril support? ";
      "Can I publish HTML to my own site instead of nytril.com, and how does that HTML do on accessibility and mobile? ";
      "If I stop using Nytril, how do I get my content out? ";
    };
    Answer: Block {
      Par {
```

```
        "Nytril has a 'Publish' feature that creates all versions of a set of documents and stores them to a local drive and
online, with one click. ";
        "The user can choose PDF, DOCX or HTML as a local output format. ";
        "Documents published online look great both on desktop and mobile devices. ";
      };
      Par {
        "Companies may wish to export locally and then use their own site for hosting. ";
        "Nytril also supports a 'white-label link' to our site. ";
        "This is a special link that writers put once on their company web site. It takes visitors to a 'Document Portal' with
your company branding and colors. ";
        "This feature allows the visitors to benefit from bookmarking, PDF download, localization and live document
walk-through. ";
      };
      Par {
        "Nytril also has a quick export feature in the IDE, that exports the currently rendered document into various document
formats, including image formats, LaTeX, SVG, Text, and more. ";
        "This allows developers to test a single document in an exported format, before publishing all documents. ";
      };
      Par {
        "The simplest way to recover content from Nytril, if moving to another system, is to publish all documents to PDF or
DOCX and import them to the new system. ";
        "If custom data structures or language tables exist, it is easy to use Nytril to loop over the data and write it out to
an external file or database. ";
        "Since Nytril is new, we have had no customers do this yet, but exporting data into usable forms is very easy. ";
      };
    };
  },

  new(#2c794b37-aa85-4256-ab3f-f72424053bd2#, "Localization") {
    Question: Span {
      "How does Nytril handle localization, measurement units, and regional paper sizes? ";
      "How does the integrated translation service work?  ";
    };
    Answer: Block {
      Par {
        "The most powerful Nytril documents are created in a way that supports text reuse and localization. ";
        "In a localized project, user-facing language strings are stored together in a separate file. ";
        "These are simply source files, but all the strings are given variable names in a " SCode("Lang"); " namespace. ";
        "For example, instead of embedding the text – "; CString("\"License Agreement\""); " – in your document, you would use a
symbol such as "; SCode("Lang.LicenseAgreement"); DotSpace;
      };
      Par {
        "A built-in feature of Nytril lets you easily gather up quoted strings from your project source files and make this
language file for you. ";
        "Another built-in feature will translate those strings to another language. ";
      };
      NumberList {
        "Choose 'Translate Language Strings' from the menu. ";
        "Choose the languages to translate into";
        "Click 'Translate'";
        "Review and edit the newly translated strings in source code";
      };
      Empty;
      Par {
        "In the end, the project will have one source file for each supported language. ";
        "Next, writers define Configuration objects in source code, to control which versions of each document they want to
publish. ";
      };
      Par {
        "A simple example is a project with two configurations: ";
      };
      NumberList {
        "English, Imperial units, Letter paper";
        "German, Metric units, A4 paper";
      };
      Empty;
      Par {
        "Once defined, Nytril will automatically publish two different versions of each document. ";
        "The correct version is shown to the customer based on their site preferences. ";
      };
      Par {
        "Nytril has full support for units and conversions between them. ";
```

```
        "Most shops will want a customized way of dealing with the display of numbers with units, and this can all be easily
handled with no limitations. ";
        "In particular, the "; DocLink("Sparling", #67eadc1b-1214-4f27-a051-de966db75067#); " example above has custom routines
for converting fractional inches to metric, in ways that handle industry specific edge cases. ";
        "The important thing is that this code is written once, it is under your control, and can be used on all the project
content, including external data. ";
      };
    };
  },

  new(#67c36905-8afd-4320-a864-571da4fe2b93#, "Workflow") {
    Question: Span {
      "How does Nytril fit into a docs-as-code workflow? ";
      "Is the source plain text that diffs cleanly in Git, and can builds run headlessly in a CI pipeline? ";
    };
    Answer: Block {
      Par {
        "Nytril source code files are plain UTF8 text. ";
        "They seamlessly operate with Git and other systems. ";
        "Here is the "; HyperLink("Nytril Template and Example", "www.github.com/Nytril/Lib");
        " repository on GitHub so you can see Nytril code in a repository. ";
      };
      Par {
        "Text review and approval is facilitated with the 'revu' string tag. ";
        revu "As an example, this string is marked for review. ";
        "It will show as purple in documents to make it easy for reviewers to identify. ";
        "A ctrl-click on the text in the document will take the reviewer to the source string for editing. ";
      };
      Par {
        "The Nytril IDE runs natively on Windows, Mac and Linux desktop systems. ";
        "As described above, it operates as its own document pipeline. ";
        "We are looking forward to getting the first request for integration into a working CI system. ";
      };
    };
  },

  new(#75a955e5-17cb-4a96-8c0d-0dae5eacace1#, "AI advantage?") {
    Question: Span {
      "One possible advantage of Nytril being a programming language is that writers might not need to learn it in depth. ";
      "They could describe what they want and let an AI coding tool write the Nytril code. ";
      "AI tools are also strong at writing code, so documents defined in code might give them more to work with than prose-based
formats. ";
      "Is that a realistic workflow today, given that Nytril is new and LLMs haven't seen much of it in training? ";
      "What are you doing to make Nytril easy for AI tools to work with (for example, reference docs, examples, or an llms.txt
";"file)?  ";
    };
    Answer: Block {
      Par {
        "You are right about the potential. ";
        "Given that Nytril allows you to specify the entire document in code, it allows an AI to mix queries, logic,
translations and formatting together to create code for the requested documents. ";
      };
      Par {
        "So far, we have been able to get LLMs to understand the content of our site. ";
        "Since our site is simply a big Nytril project, this means that users who publish to the site will benefit from having
their content published in a way that is digestible to Google and AI crawlers. ";
      };
      Par {
        "As far as AI generated Nytril code, we have had success there as well. ";
        "The following diagram was generated by ChatGPT after reading our ";
        DocLink("tutorial document", "fda46c87-c829-4c9d-9d64-c89b07aa6643/85");
        ", and being asked to produce a drawing of a machine in Nytril code. ";
        "The generated source for this diagram is at the very end of this document. ";
      };
      Par {
        ParAlignment: Center;
        ProductMachine(1");
      };
      Par {
        "It is early days on the LLM front, but we foresee that AI systems will be able to deliver documents in 'source form'
rather than as DOCX or PDF files. ";
        "This is vastly more useful for content creators. ";
```

```
        "It will allow writers to edit and correct the inevitable hallucinations, and use that work as a permanent resource for
future documents. ";
      }
    };
  },

  new(#be8a2096-fe3c-4b68-aa8b-38052643ed54#, "Pricing") {
    Question: Span {
      "Who is using Nytril now, how does pricing work between the Community and Professional editions, and how should a team
think about adopting a tool from a small, new company? ";
    };
    Answer: Block {
      Par {
        "The Community Edition can be downloaded off the site for free and used by anyone. ";
        "It installs in less than 1 minute. ";
        "Please try it out. ";
      };
      Par {
        "For corporate teams, we welcome the use of the Community Edition for evaluation. ";
        "The subscription is $9 per month to enable the Pro Edition. ";
        "With a Pro Edition, teams will have access to translation and publishing/hosting services. ";
      };

      Par {
        "Nytril is newly released, but has been in development and beta testing for years. ";
        "It was very important to us that the product was polished before we started to spread the word. ";
        "We do not have a lot of users yet, but the ones that we do have are enthusiastic, and provide great feedback. ";
        "Nytril LLC is a small company, and therefore reacts rapidly to user requests and bug reports. ";
        "Giving writers power to create is our mission. ";
      };

      Par {
        "Becoming an early adopter of new tech is an important decision for a team. ";
        "It comes down to whether the increased productivity and/or lower cost of the new product is a greater benefit than the
switching cost over some reasonable timeframe. ";
        "I believe that once writers try it out, and understand the productivity and flexibility gains from this docs-as-code
model, they will make the switch with enthusiasm and wonder how they got along without it. ";
      };
    };
  },
];

//========================================================================
// AI Generated source code for a drawing of a machine, as discussed
//========================================================================

ProductMachine(height, lineColor=Colors.Black, ratio=1.35) = Canvas(height * ratio, height) {
  Baseline: 90%;

  var line = new Stroke(3pt, lineColor, LineStyles.Solid, LineCaps.Round, LineJoins.Round);

  // Base
  Shape(null, line) {
    ClosedPath(13%, 87%) {
      LineTo(87%, 87%);
      LineTo(87%, 95%);
      LineTo(13%, 95%);
      LineTo(13%, 87%);
    };
  };

  // Left support
  Shape(null, line) {
    OpenPath(29%, 77%) {
      LineTo(29%, 87%);
      LineTo(39%, 87%);
      LineTo(39%, 77%);
    };
  };

   // Right support
  Shape(null, line) {
```

```
  OpenPath(70%, 77%) {
    LineTo(70%, 87%);
    LineTo(80%, 87%);
    LineTo(80%, 77%);
  };
};

// Left rear rounded housing
Shape(null, line) {
  ClosedPath(20%, 40%) {
    CurveTo(new(14%, 40%), new(10%, 45%), new(10%, 53%));
    LineTo(10%, 68%);
    CurveTo(new(10%, 76%), new(14%, 80%), new(20%, 80%));
    LineTo(25%, 80%);
    LineTo(25%, 40%);
    LineTo(20%, 40%);
  };
};

// Main motor housing
Shape(null, line) {
  ClosedPath(25%, 40%) {
    CurveTo(new(25%, 37%), new(27%, 36%), new(30%, 36%));
    LineTo(55%, 36%);
    CurveTo(new(58%, 36%), new(60%, 38%), new(60%, 41%));
    LineTo(60%, 75%);
    CurveTo(new(60%, 78%), new(58%, 80%), new(55%, 80%));
    LineTo(30%, 80%);
    CurveTo(new(27%, 80%), new(25%, 78%), new(25%, 75%));
    LineTo(25%, 40%);
  };
};

// Small box top of motor
Shape(null, line) {
  ClosedPath(31%, 29%) {
    CurveTo(new(31%, 27%), new(32%, 26%), new(34%, 26%));
    LineTo(49%, 26%);
    CurveTo(new(51%, 26%), new(52%, 27%), new(52%, 29%));
    LineTo(52%, 36%);
    LineTo(31%, 36%);
    LineTo(31%, 29%);
  };
};

// Connection between motor and right housing
Shape(null, line) {
  ClosedPath(60%, 45%) {
    LineTo(68%, 45%);
    LineTo(68%, 72%);
    LineTo(60%, 72%);
    LineTo(60%, 45%);
  };
};

// Right square housing
Shape(null, line) {
  ClosedPath(68%, 39%) {
    CurveTo(new(68%, 36%), new(70%, 35%), new(73%, 35%));
    LineTo(84%, 35%);
    CurveTo(new(87%, 35%), new(89%, 37%), new(89%, 40%));
    LineTo(89%, 75%);
    CurveTo(new(89%, 78%), new(87%, 80%), new(84%, 80%));
    LineTo(73%, 80%);
    CurveTo(new(70%, 80%), new(68%, 78%), new(68%, 75%));
    LineTo(68%, 39%);
  };
};

// Four bolt heads
Shape(lineColor) {
  EllipsePath(new(new(73%, 42%), new(2.5%, 2.5%)));
```

```
    EllipsePath(new(new(83%, 42%), new(2.5%, 2.5%)));
    EllipsePath(new(new(73%, 69%), new(2.5%, 2.5%)));
    EllipsePath(new(new(83%, 69%), new(2.5%, 2.5%)));
  };

  // Vertical risers to upper tank
  Shape(null, line) {
    OpenPath(75%, 35%) {
      LineTo(75%, 27%);
    };
    OpenPath(80%, 35%) {
      LineTo(80%, 27%);
    };
  };

  // Upper tank
  Shape(null, line) {
    ClosedPath(69%, 7%) {
      CurveTo(new(69%, 4%), new(71%, 3%), new(74%, 3%));
      LineTo(88%, 3%);
      CurveTo(new(91%, 3%), new(93%, 5%), new(93%, 8%));
      LineTo(93%, 24%);
      CurveTo(new(93%, 27%), new(91%, 29%), new(88%, 29%));
      LineTo(74%, 29%);
      CurveTo(new(71%, 29%), new(69%, 27%), new(69%, 24%));
      LineTo(69%, 7%);
    };

    OpenPath(69%, 16%) {
      LineTo(93%, 16%);
    };
  };

  // Pipe from motor top to tank
  Shape(null, line) {
    OpenPath(41%, 26%) {
      LineTo(41%, 16%);
      CurveTo(new(41%, 12%), new(44%, 11%), new(47%, 11%));
      LineTo(65%, 11%);
    };
  };

  // Small connector before tank
  Shape(null, line) {
    ClosedPath(65%, 8%) {
      LineTo(69%, 8%);
      LineTo(69%, 19%);
      LineTo(65%, 19%);
      LineTo(65%, 8%);
    };
  };
};
```