



CONVERTING LEGACY UNSTRUCTURED CONTENT TO DITA WITH AI

Or, Converting Old Car Manuals to XML
Using Artificial Intelligence

KEITH SCHENGILI-ROBERTS

DITAWRITER.COM

WHO IS THIS GUY?

- Keith Schengili-Roberts, Head DITAWriter of DITAWriter.com
- Started as a Technical Writer 30+ years ago, currently working as Senior Staff Content Engineer at Teradata
- Have worked primarily as a Content Strategist and Consultant since early 2000s
- DITAWriter.com is my consulting company, also an industry blog established 15 years ago
 - I specialize in helping companies move from unstructured to structured (DITA) content
 - You can contact me at: keith@ditawriter.com



Selfie with SF theme generated by AI Profile Pic Maker

AGENDA

- Converting unstructured content using AI
- Why the need for DITA example files?
- About the Model T car manual
- The process, and the many pros and cons discovered along the way when using an AI to convert unstructured content
- Things the human (me) still had to do
- Using AI to create reusable content
- Copyright issues relating to AI-generated content
- AI benefits when using structured content?
- Final thoughts + Q/A

CONVERTING UNSTRUCTURED CONTENT USING AI

- Most of the focus on Artificial Intelligence (AI) has been on using it for Chatbots
- The idea of using AI and Large Language Models (LLMs) to produce structured content directly seems to have been largely overlooked
 - We are seeing tools (like Oxygen's AI Positron Assistant or Intuillion's DITA.ai) designed to assist technical writers, but very little of what I have seen talked about the hands-on experience, specifically when working with DITA
 - So I decided to dive in myself and share my experiences
 - There are also some copyright issues relating to AI-generated text that I will get to later
- There are a lot of pros and cons
- The benefits are there, but there is still work to be done...



WHY THE NEED FOR DITA EXAMPLE FILES?

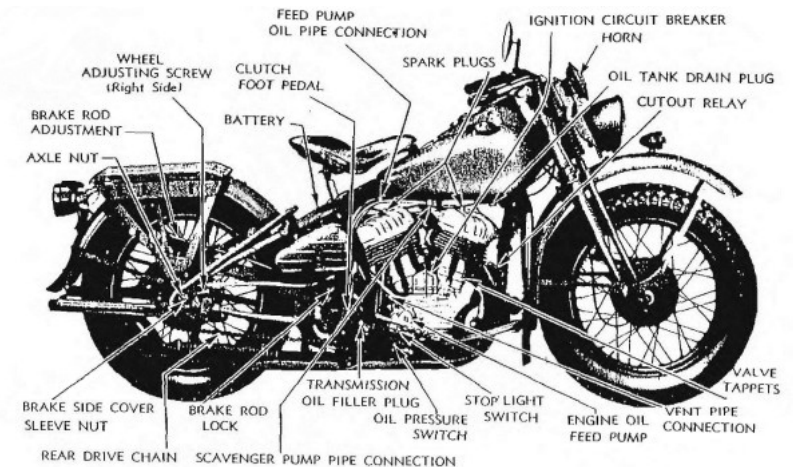
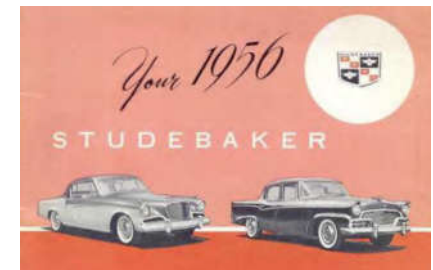
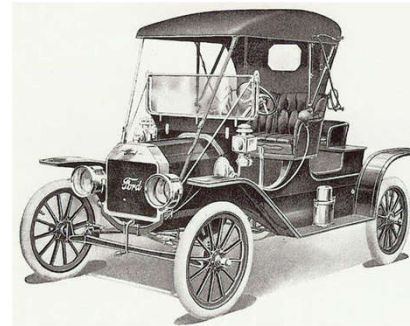
- There are few publicly-available sets of DITA sample code for people to learn from
 - Examples that come with the DITA specifications are sparse, and the intended audience is for developers, not technical writers
 - Easier for technical writers and students to learn best practices by looking at (and playing with) example code
 - Recently I have been working with the ACM SIGDOC Structured Authoring Committee (acm-sigdoc-structured.org/) and example code is one of the things instructors want
- Have posted several examples of DITA-fied manuals from out-of-copyright materials available at: github.com/DITAWriter
 - With help from DITA Adoption Committee members, we converted a WWII Pilot Training Manual for the Mitchell B-25 Bomber to DITA
 - Other examples show the ways to implement DITA keys, glossaries, and LwDITA / MDITA sample code

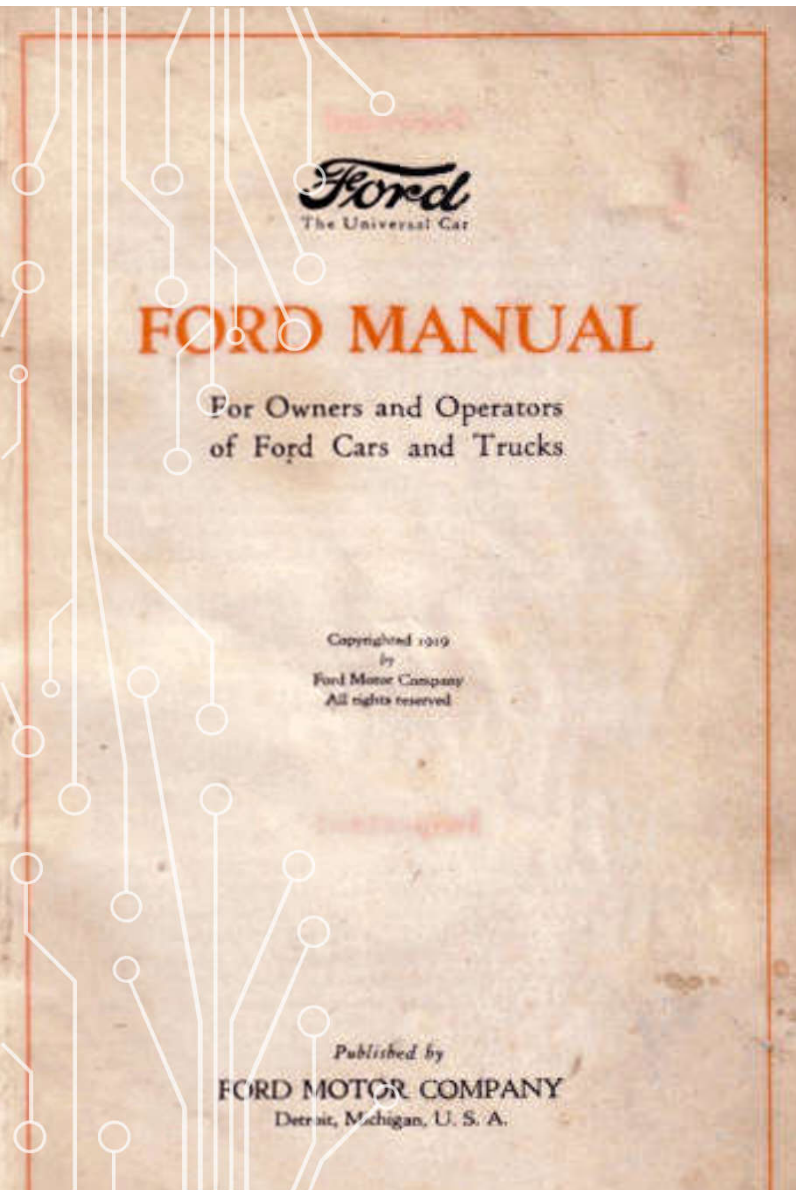
The screenshot shows the GitHub profile for 'DITAWriter'. The profile includes a circular profile picture of a man with glasses and a beard. Below the picture is the name 'DITAWriter' and a 'Follow' button. The profile statistics show '12 followers' and '0 following'. There are sections for 'Achievements' and 'Block or Report'. The main content area is titled 'Popular repositories' and lists several repositories:

- LwDITA_Code_Samples** (Public): A set of code samples designed to demonstrate what is possible for each of the "flavors" of Lightweight DITA. It has 11 stars and 12 forks.
- pilot_training_mitchell_bomber**: A DITA-fied version of the Pilot Training Manual Mitchell Bomber B-25. It has 5 stars and 3 forks.
- dita_keys_examples** (Public): A set of DITA examples displaying how keys can be deployed in a step-by-step fashion. Each directory contains a version of a DITA-fied manual for the TRS-80 Expansion Pack, a long out-of-copyright... It has 2 stars and 1 fork.
- elements_of_style_Strunk** (Public): A DITA-fied version of Strunk's original Elements of Style, derived from an out-of-copyright version published in 1920. It has 1 star.
- MDITA_GarageSampleCode**: MDITA (Markdown DITA) sample code based on familiar DITA "Garage Example" sample code. It has 1 star.
- Model_T_Manual_AI_DITA_Conversion**: It has 1 star.

CHOOSING THE CONTENT TO CONVERT

- Big fan of Project Gutenberg (www.gutenberg.org), which holds several out-of-copyright manuals, including:
 - Ford Manual for Owners and Operators of Ford Cars and Trucks (1919)
 - Motorcycle, Solo (Harley-Davidson Model WLA) (1943)
- The Internet Archive (archive.org) also had some useful material:
 - Studebaker owner manuals for 1948 and 1956
 - These are versioned so a good source for investigating content reuse and AI
- Why vehicle manuals?
 - Easily relatable for most people, and the manuals are mostly “modern” in style



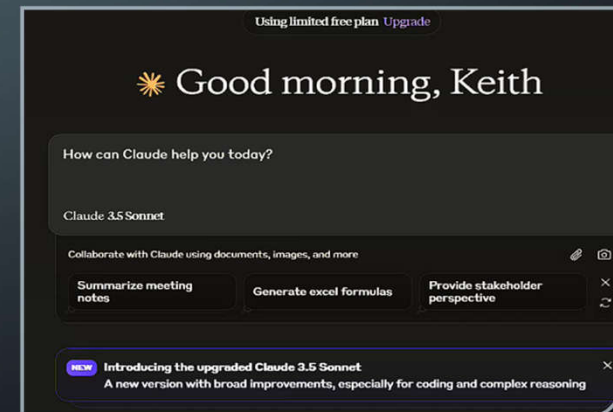


ABOUT THE MODEL T MANUAL

- The original Ford Manual from 1919 was written as a series of frequently asked questions (FAQs)
- Most of the questions are straightforward:
 - How is the Car started?
 - What causes “Knocking” in the Engine?
- Also some oddly-phrased questions:
 - When the Valves and Push Rods are worn—what?
 - If the Coil and Plug are right—what?
 - How is a Weak Unit detected?
- The dangling “what” at the end is shorthand for “what to do?” or “what should be done?”
- In true narrative fashion for these old manuals, each FAQ question builds on the previous one, so the “weak unit” question relates to the answer to the previous question on “coil units”

THE PROCESS

- After testing several AI's, I found Claude.AI from Anthropic to produce the best results
 - I find that this is still the case, but have switched to ChatGPT for more recent conversion efforts for reasons I will cover later
- Process: I copied a topic's worth of content from an HTML version of the original manual and asked Claude to convert it to a specific DITA topic type, while also asking it to provide a short description.
- Typical prompt: "Please convert the following to a DITA concept topic, including a short description: [pasted text from source]".
 - In a few cases, I asked Claude to choose the appropriate DITA topic type for the pasted content, but the results were mixed; I found it easier to guide the AI with the topic typing
- Copied completed topic code into Oxygen for validation/checking



PRO: CONVERSION IS FAST

Claude

Using limited free plan [Upgrade](#)

☀️ Good morning, Keith

How can Claude help you today?

Claude 3.5 Sonnet

Collaborate with Claude using documents, images, and more

Summarize meeting notes Generate excel formulas Provide stakeholder perspective

NEW Introducing the upgraded Claude 3.5 Sonnet
A new version with broad improvements, especially for coding and complex reasoning

Your recent chats View all →

- Starting Engine in Cold Weather
10 seconds ago
- Ford Car Ownership and Maintenance Guide
1 day ago
- Troubleshooting Excessive Exhaust...
1 day ago
- Troubleshooting Red Signal Light Failure
1 day ago
- Engine Compression Test Procedure
1 day ago
- DITA Troubleshooting Topics for Engine Issues
1 day ago

K

PRO: CONVERSION IS FAST

- As you just saw, Claude immediately started the process of converting unstructured content given to it into DITA code
- The longest topics took no more than 20-30 seconds
- Claude was also good at taking an oversized “supertopic” source (such as an extended troubleshooting section containing multiple causes and remedies) and breaking it down into smaller, more concise topics.
- The speed of conversion is a net positive even in cases where the results needed tweaking by a human (me)

K Can you break this into multiple troubleshooting topics, one for each condition and a separate troubleshooting solution?

Yes, I can break this into separate troubleshooting topics. Here's the first one as an example - let me know if you'd like me to do the others as well.

Panel Lamps Troubleshooting
Click to open document

Would you like me to create the separate topics for:

1. Dim lights that brighten during acceleration
2. Lights exceeding normal brilliance during acceleration
3. Blackout/service stop lamp failure
4. Horn not sounding?

Each would follow the same structure but with its specific condition and troubleshooting content.

K Please create all of the separate troubleshooting topics

I'll create individual troubleshooting topics for each issue.

Panel Lamps Troubleshooting
Click to open document

Dim Lights Troubleshooting
Click to open document

Excessive Light Brilliance Troubleshooting
Click to open document

CON: LOTS OF BACK-AND-FORTH

- There were many times when the DITA code created did not validate.
 - I would ask it to re-validate its code to see if it could figure out where it went wrong.
 - Often it was able to figure it out, but there were several times where it would come up with substitute code that was incorrect in a different way.
- Troubleshooting topics were particularly problematic for Claude, often taking several runs before getting it right.
 - For many of these topics it was easier for me to correct the code myself.

K There are two sets of steps tags in your example code

Ah yes, you're right! In DITA, we should use a single `steps` element with nested `stepsection` elements, or use `steps-unordered` if the order isn't critical. Let me correct that.

</>

Validated Motorcycle Stop and Park DITA Task
Click to open code

Changes made:

1. Removed the second `steps` element
2. Kept all steps within a single `steps` element
3. Used `stepsection` to separate and label the two procedures

This structure now better follows DITA task topic requirements. Would you like me to explain anything else about the DITA structure?

K You cannot have a `ul` tag between a `howtoavoid` tag, and there is also an `info` tag inside a `stepresult`.

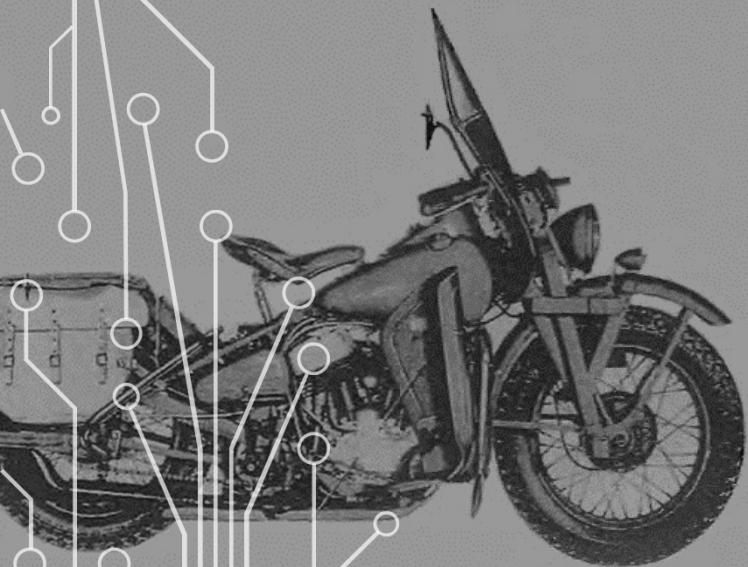
Ah yes, you're absolutely right! In DITA, `howtoavoid` should contain only basic text or `ph` elements, and the structure of `info` and `stepresult` needs to be corrected. Let me fix that.

TECHNICAL MANUAL
9-879

WAR DEPARTMENT

Washington, 18 October 1943

MOTOR CYCLE, SOLO (Harley-Davidson Model WLA)



PRO: AI EXCELS AT RE-WRITING

- I took a stab at converting this document myself several years ago and gave up because it is such a bugbear.
 - I find this a poorly-written technical document: it meanders, it is pedantic and is the opposite of concise.
 - For example, it lists 86 things to check for preventative maintenance.
- I stopped using this as a source for conversion as I wasn't happy with the results, but Claude does not complain when given large chunks of this content to digest, and its rewrites were concise.
- It does not reproduce the content word-for-word (this could be a problem in some circumstances), but instead intelligently restructures and rewrites the content to make it more easily digestible.

CON: AI “HALLUCINATIONS”

- From an AI perspective, “hallucination” is when an AI creates a response that contains false information; in this case problematic code rather than content
 - Likely due to modeling on multiple programming languages
- Examples: there were many occasions when the AI would create tags out-of-the-blue or invalid structures
 - Easily spotted during validation testing
- A human needs to be “in the loop” to validate code and check content veracity

K In the previous code example, what is the `refelem` tag?

I apologize for using the non-standard `refelem` tag. You're right to question this - it's not a valid DITA tag. Let me correct the reference topic to use proper DITA elements.

K The code sample has nested task tags

Ah yes, I apologize. Let me revise this to use `steps-unordered` and sections within a single task instead of nested tasks, which is better DITA practice.

K You cannot have a `ul` tag between a `howtoavoid` tag, and there is also an `info` tag inside a `stepresult`.

Ah yes, you're absolutely right! In DITA, `howtoavoid` should contain only basic text or `ph` elements, and the structure of `info` and `stepresult` needs to be corrected. Let me fix that.

PRO (MIXED): AI CAN ACT AS A SME

- One time I asked Claude to convert some content that contained some obviously dated car terminology, so the AI came back with converted text that mentioned that the process applied only to vintage vehicles.
 - I fixed this but had to keep on the lookout for these types of unwelcome additions.
- Later, I wanted to include a glossary of terms, which did not exist in the original Model T manual
 - Reason for straying from the original: there are few good examples of working glossaries in DITA
- I asked Claude to create individual DITA glossentry topics, adding the context that it was about Model T car parts
- In this case Claude acted as a Subject Matter Expert (SME), creating new content from its store of knowledge


Starting a Car

Short Description: Learn the precise steps to start and operate a manual transmission vehicle from the early automotive era.

Bendix Drive

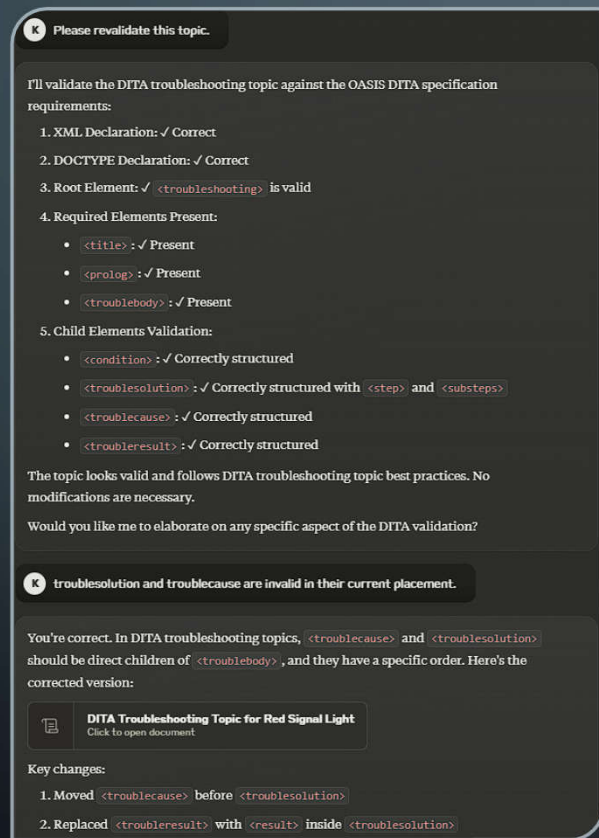
A mechanical device in the Model T Ford's starting system that automatically engages and disengages the starter motor pinion gear with the engine's flywheel ring gear. The Bendix drive uses inertia and a spiral-cut drive sleeve to temporarily couple the starter motor to the engine during starting, then automatically disengages once the engine begins running.

Bendix drive

 The term is commonly found in Model T electrical system documentation, repair manuals, and starter motor specifications. It represents a significant advancement in self-starting technology for early automobiles.

Bendix starter drive

CON: CAN'T RELY ON SELF-VALIDATION



- Related to the previous example, while you can ask the AI to re-validate its own content, that doesn't mean that it actually does so
 - In this example, it supposedly validates troubleshooting, troubleresult, and troublecause, but "troubleSolution" is in camel-caps, and there are no such tags as "troubleresult" or "troublecause" (there are "remedy" and "cause" tags)
 - So clearly the AI does not run a true validation pass, and has a tendency to make things up
 - This can be overcome by feeding the DITA DTD to the AI and tell it to validate against that
 - This depends in part on how much content can be used in the prompt process in addition to the content being converted (character count limit)

CON: CONSISTENCY IS NOT A STRENGTH

- I was using the free instance of Claude, which only allowed a certain number of questions per session/day
- Claude would consistently use one style within a single session, but if I had to start a new session, the previous context was lost.
- The contrast was most obvious in the “Summary of Engine Troubles and Their Causes” chapter where the first series of topics were done in a complex, multi-faceted manner, and the rest—done in a new session, but using the same prompts—came up with a stripped-down version of the same type of material.
- This can be overcome by having a dedicated session.

Engine Start Failure Conditions
Common conditions that can prevent the engine from starting properly.

Fuel System Issues

- Gas mixture too lean
- Water in gasoline
- Gasoline supply shut off
- Carburetor frozen (in zero weather)
- Water frozen in gasoline tank sediment bulb

Electrical System Issues

- Vibrators adjusted too close
- Water or congealed oil in commutator
- Magneto contact point (in transmission cover) obstructed with foreign matter
- Coil switch off

Environmental Factors

! Important:
Several conditions are specifically related to cold weather operation:

- Carburetor freezing at zero temperatures
- Frozen water in the sediment bulb
- Congealed oil in the commutator

Full-fledged Topic

Engine Knocking
Common causes of engine knocking and their identification.

Causes

- Carbon deposits accumulated on piston heads
- Connecting rod bearing has become loose
- Crankshaft bearing has become loose
- Spark timing advanced beyond specification
- Engine operating above normal temperature range

Stripped-down Topic

PRO: CAN EASILY AUTOMATE REPETITIVE ACTIONS

- While the AI may not have an inherent understanding of the topics it creates, you can ask it to structure a basic bookmap for you
 - You provide the structure and give it the order of the filenames
- You can also ask it to replace the topicrefs with conkeyrefs and have those keys placed into a separate file that is referenced by the bookmap
 - Given some guidance, an AI can automate conversion processes like this easily and quickly

Claude DITA Bookmap for Car Operation

Please put together the following DITA topics in the following order within a DITA bookmap: the_car_and_its_operation.dita as a chapter, followed by these sub-topics: prepare_radiator_task.dita, gasoline_tank_preparation_task.dita, engine_oil_maintenance_task.dita, spark_throttle_operation_task.dita, and engine_start_lever_settings_reference.

```
</> DITA Bookmap for Car Operation
Click to open code

I've created a DITA bookmap that structures the topics as you specified, with "the_car_and_its_operation.dita" as the primary chapter, and the other topics nested as its child topics.
```

```
<?xml version="1.0" encoding="UTF-8"?>
<DOCTYPE bookmap PUBLIC "-//OASIS//DTD DITA Bookmap V1.2//EN" "bookmap.dtd">
<bookmap>
  <title>Car operation and maintenance guide</title>
  <chapter href="the_car_and_its_operation.dita">
    <topicref href="prepare_radiator_task.dita"/>
    <topicref href="gasoline_tank_preparation_task.dita"/>
    <topicref href="engine_oil_maintenance_task.dita"/>
    <topicref href="spark_throttle_operation_task.dita"/>
    <topicref href="engine_start_lever_settings_reference.dita"/>
  </chapter>
</bookmap>
```

Claude DITA Bookmap for Car Operation

I've created a DITA bookmap that structures the topics as you specified, with "the_car_and_its_operation.dita" as the primary chapter, and the other topics nested as its child topics.

Can you restructure this as DITA conkeyrefs?

```
</> DITA Conkeyref Bookmap for Car Operation
Click to open code

I've restructured the bookmap to use key references (conkeyrefs) for each topic, defining keys for each document and then referencing those keys in the chapter structure.
```

Could you place the conkeyrefs in a separate, referred file?

I'll create two artifacts: one for the keys file and one for the bookmap.

```
</> DITA Keys File for Car Operation
Click to open code

</> DITA Bookmap Referencing Keys
Click to open code

I've separated the keys into a distinct car_operation_keys.dita file and modified the bookmap to reference this keys file using <conkeyref>. This approach allows for more modular and reusable key definitions.
```

```
<?xml version="1.0" encoding="UTF-8"?>
<DOCTYPE bookmap PUBLIC "-//OASIS//DTD DITA Bookmap V1.2//EN" "bookmap.dtd">
<bookmap>
  <title>Car Operation and Maintenance Guide</title>
  <conkeyref href="car_operation_keys.dita"/>
  <chapter keyref="car_operation">
    <conkeyref keyref="radiator-prep"/>
    <conkeyref keyref="gas-tank-prep"/>
    <conkeyref keyref="engine-oil-maint"/>
    <conkeyref keyref="spark-throttle"/>
    <conkeyref keyref="start-lever-settings"/>
  </chapter>
</bookmap>
```

THINGS THE HUMAN (ME) HAD TO DO

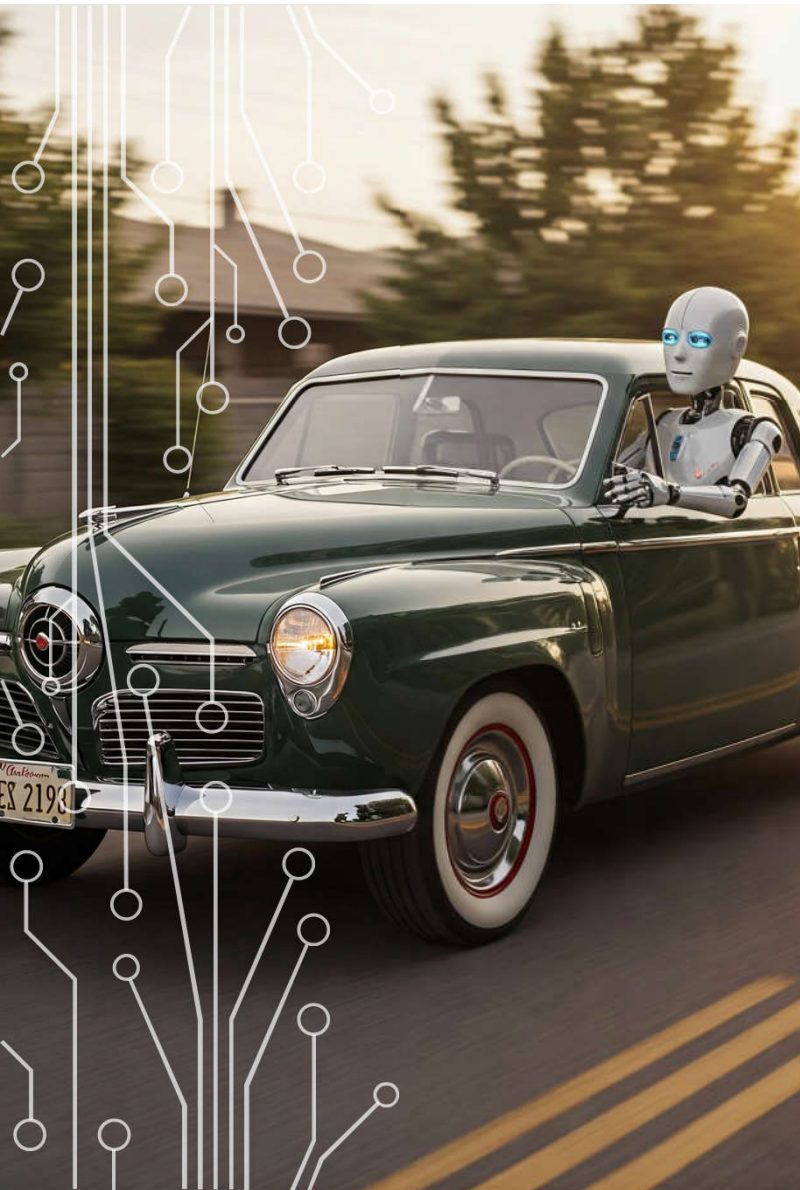
- Validate the existing code
- Check that the re-written content made sense
- The source was HTML-based, and the AI could not “see” any of the images; these had to be inserted by hand
- I also added relationship tables between the topics, added the glossary as backmatter, and structured the bookmap to match that of the original
- A human needs to be in the loop to check that the material is human-understandable



OUTCOME: ONE DITA-FIED MODEL T MANUAL

- Converted Model T Manual consists of 204 files
 - 179 topics
 - 25 images
 - 1 ditamap, plus 3 sub-maps (keydef, image store, and glossary)
 - All available for free at github.com/DITAWriter/Model_T_Manual_AI_DITA_Conversion
- As it is not a word-by-word copy of the original manual, I consider this a “version” or “variant” of the original



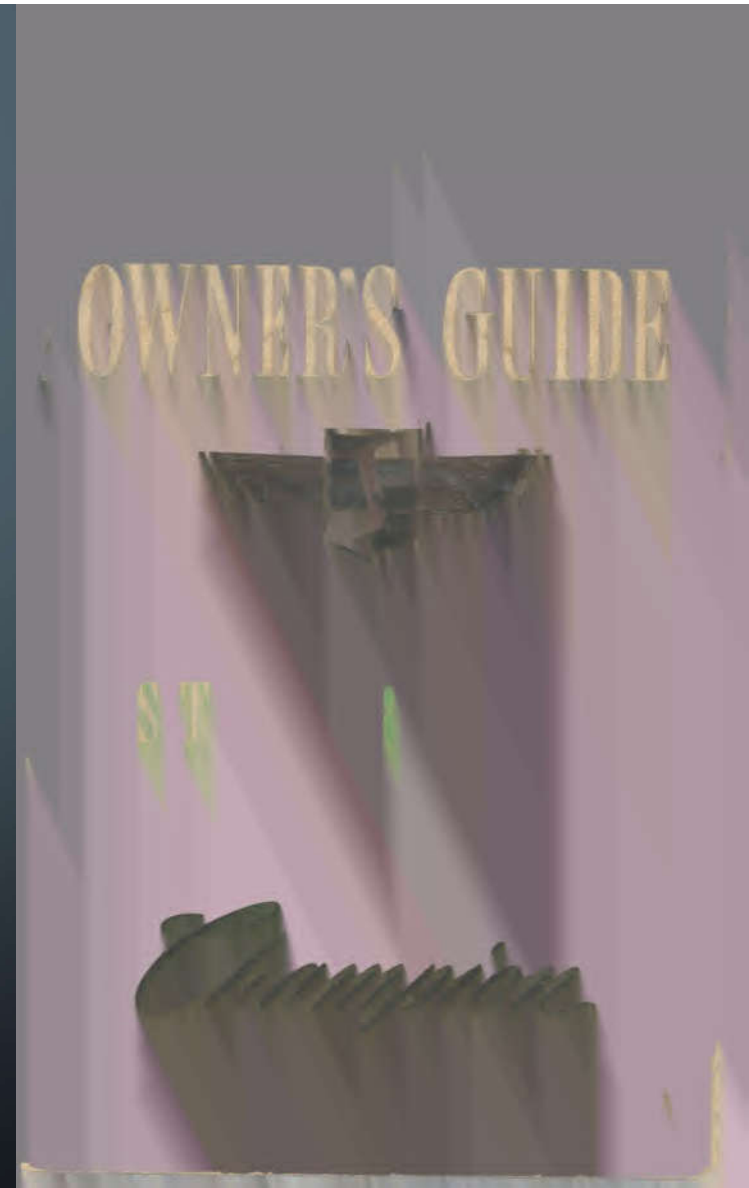


DITA-BASED CONTENT REUSE AND AI

- Working with Studebaker car manuals from 1948 and 1956 that share similar content
- This is continuing work, investigating how well AI (ChatGPT 5 in this case) can work within a single session identifying and creating reusable content between versioned manuals
 - Why ChatGPT 5? I needed to purchase a plan that would allow me to process multiple large documents. Also wanted to see what the latest version of ChatGPT had to offer.
 - Ran into the same issues with validation as before.

WORKING WITH PDFS AND IMAGES

- The content from archive.org comes as PDFs with embedded text using optical character recognition (OCR)
 - OCR is still far from perfect, and applying AI to it doesn't help much
- Content converted from PDF to Word
 - Word to DITA is a more common use-case for technical writers
 - Found that ChatGPT 5 cannot usefully extract images from the PDFs; it sees each page as an image and does what it can (see typical example to the right)
 - It can do a good (not great) job at comparing image use between the manuals; needs to be almost exact to work
 - It does better with the Word file from a converted PDF, but it definitely must be checked by a human
 - In the end, many images were extracted by hand



AI AND CREATING SHARABLE CONTENT

- Turns out ChatGPT5 is very good at determining textual content that is *similar enough* to determine what content is potentially sharable
- Interestingly, it did better at this with the Word-based content than with the already DITA-fied content
- The approach that worked best was to mix the best of both worlds: first getting ChatGPT 5 to analyze the Word content, and then using that analysis as a guide on crafting the sharable content in DITA

Awesome brief. I compared the 1948 and 1953 Studebaker owner's guides and mapped what you can safely share as common DITA building blocks—both at the topic level (whole topics you can reuse) and paragraph/snippet level (conrefs you can point to from year-specific topics).

What's sharable at the topic level (DITA topic reuse)

Below are topics that are either identical or close enough to be turned into a single "common" topic with light key-driven wording (model/year) and conditionalized details where needed:

1. Warranty & policy boilerplate

- Factory warranty text is essentially the same (90-day/4,000-mile, exclusions, "in lieu of all other warranties"). Put this in one **concept** topic (e.g., `warranty.dita`) with brand-generic wording.

`studebaker_owners_guide_1948_te...` `studebaker_champion_1953_owners...`

- Battery warranty prose also lines up (Willard, prorated adjustment). Make it a subsection or a sibling

`battery-warranty.dita` `studebaker_owners_guide_1948_te...` `studebaker_champion_1953_owners...`

- Dealer/service policy overview is conceptually identical; consolidate as `dealer-service-policy.dita`.

`studebaker_owners_guide_1948_te...` `studebaker_champion_1953_owners...`

2. Instruments & controls overview

The list of primary controls (ignition switch, starter, ammeter, heat/temperature, oil pressure, lights, beam switch, parking brake, clutch, gearshift, wiper/washer, seat adjuster, hood release) and the way they're explained follow the same pattern and phrasing. Use one **concept** topic `instruments-controls.dita` with conkeyrefs for any diagram callouts and minor year/name differences (e.g., "map light" vs. "instrument light").

`studebaker_owners_guide_1948_te...` `studebaker_champion_1953_owners...`

3. Break-in / Run-in guidance

Both books prescribe sub-60 mph limits for the first ~1000 miles and discourage pumping the accelerator; the editorial stance is the same. Create one **concept** topic `break-in.dita` keyed for distances/thresholds.

`studebaker_owners_guide_1948_te...` `studebaker_champion_1953_owners...`

4. Operating the car (start/shift/stop)

- Cold/hot starting, "do not pump," neutral check, warm-up to ~170°F, oil-pressure check. One **task** topic

`start-engine.dita` `studebaker_owners_guide_1948_te...` `studebaker_champion_1953_owners...`

- Manual shifting best practices (don't jump gears, stop before reverse). One **task** `shift-gears.dita`.

`studebaker_owners_guide_1948_te...` `studebaker_champion_1953_owners...`

- Push/tow starting contingency is the same (OD locked out). Fold into the start task as an **optional** step or a separate short task.

`studebaker_owners_guide_1948_te...` `studebaker_champion_1953_owners...`

5. Overdrive

The engagement logic (cut-in around ~30–35 mph, kickdown to conventional, lockout on grades/tow) is



+ Ask anything

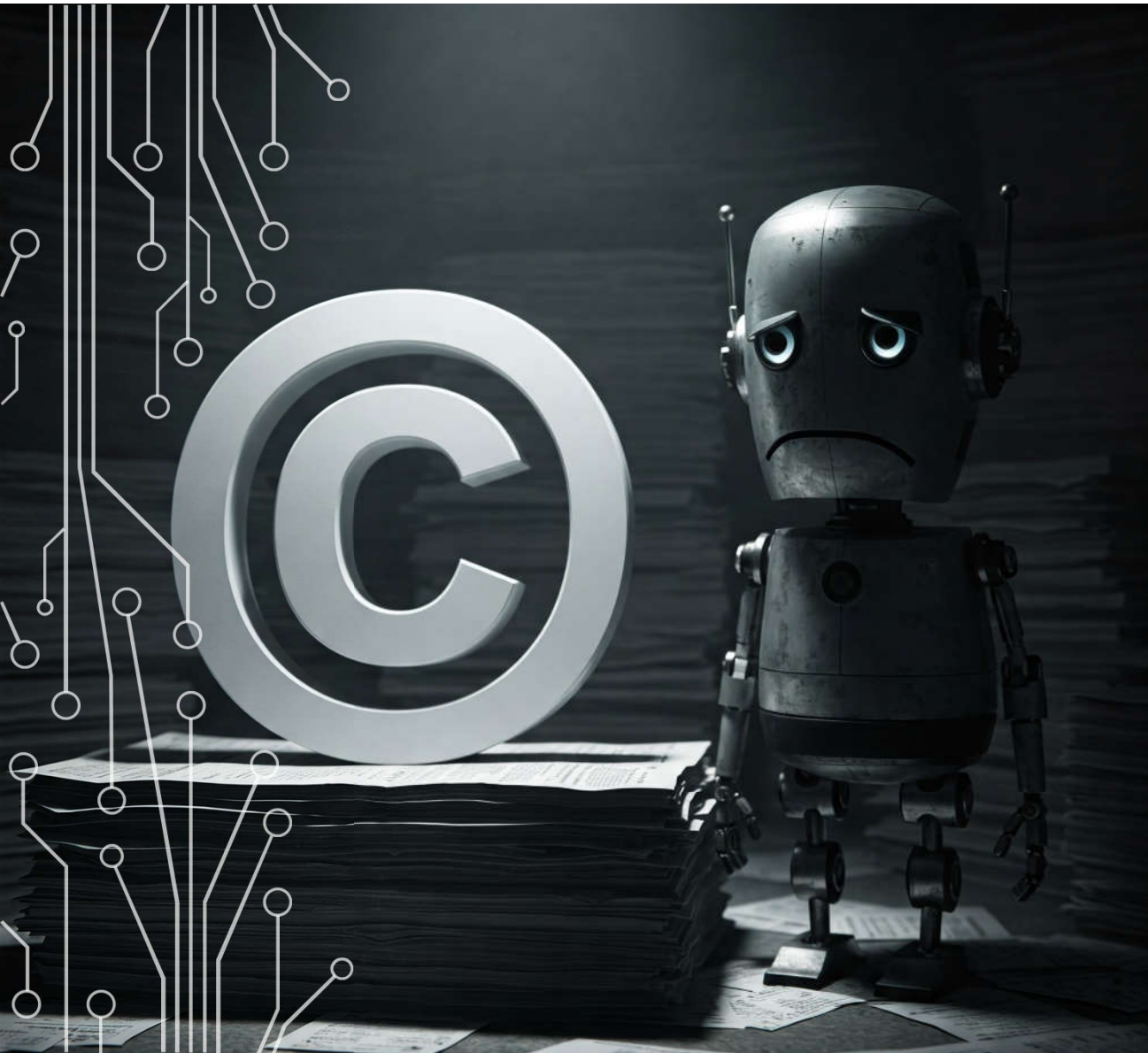


ChatGPT can make mistakes. Check important info. See [Cookie Preferences](#).



ONGOING WORK ON CREATING SHARABLE CONTENT

- ChatGPT 5 appears to be good at determining reusable content at the word/phrase (keys), paragraph and topic levels (conkeyrefs)
- Hoping to come up with a version of these manuals that includes DITA scoped keys using AI



COPYRIGHT AND AI-GENERATED CONTENT

- At the moment, within the U.S., only a human can be considered as an author of a copyrightable work.
- A recent U.S Copyright Office statement:
 - “Based on the Office’s understanding of the generative AI technologies currently available, users do not exercise ultimate creative control over how such systems interpret prompts and ... the generated material is not the product of human authorship. As a result, that material is not protected by copyright and must be disclaimed in a registration application”

SUI GENERIS AND AI + HUMAN MEDIATED CONTENT

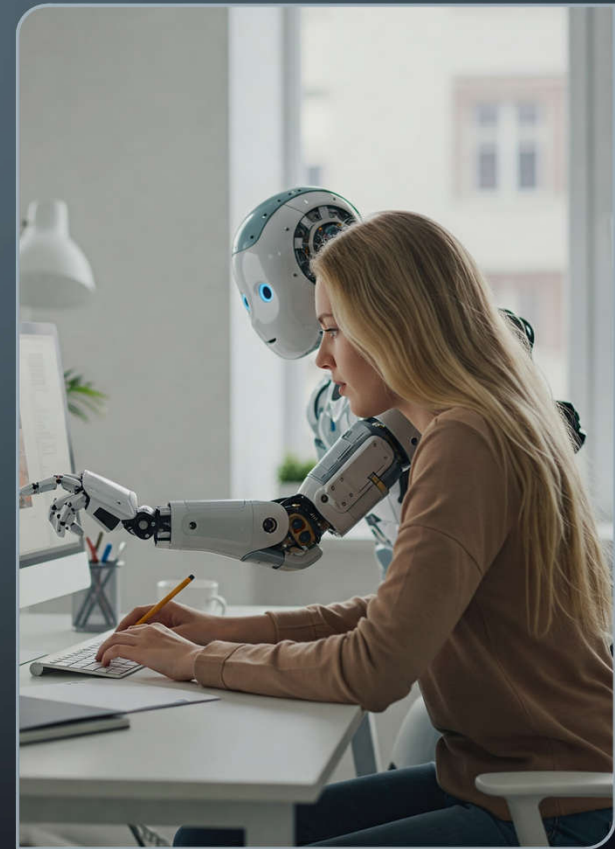
Sui generis ("of its own kind") legal may offer a solution for protecting AI-generated content

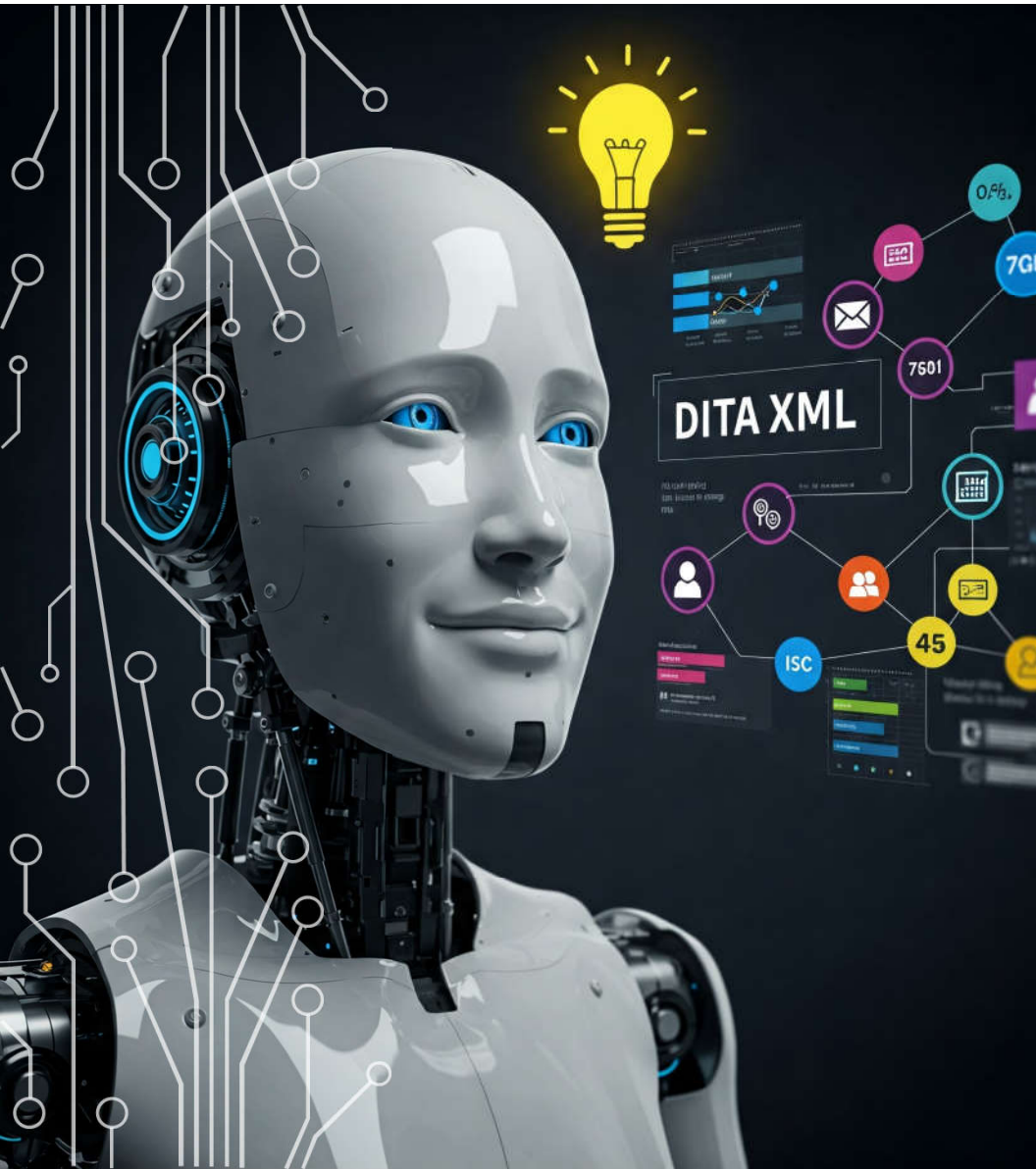
- UK offers a provision in their copyright act for "computer-generated works" where copyright belongs to the person who co-created the work; a form of sui generis
- Ukraine has amended its laws to introduce a specific sui generis right for AI created content belonging to the software licensee



COPYRIGHT IS SHAPING OUR INTERACTION WITH AI

- What I am seeing in the market are not applications that are trying to “take over” and create content directly, but instead work with people to enhance or work alongside them
- For example, while ChatGPT is perfectly capable of generating full software code, it is being seen more in software development platforms as a co-development tool used by programmers
- Adobe Photoshop is incorporating AI-based features such as “generative fill”, which can add, remove, or expand upon existing image content
- Similarly, AIs are not being asked to write content from scratch, but instead interact with the user in a prompt-based situation as a classic chatbot; there is no copyright applicable to searches





AI BENEFITS OF USING STRUCTURED CONTENT?

- There are claims that structured content can produce better chatbot responses
 - Does not appear to be an “out of the box” thing; it is not instantly better
 - Worth keeping in mind that LLMs are trained on unstructured content
- Retrieval-Augmented Generation (RAG) appears to be a necessary step, combining knowledge graphs (which provide context) to structured content
 - A knowledge graph is essentially a graph database that stores information as a network of interconnected nodes and relationships, essentially an ontology applied to a set of data/structured content
 - This provides a deeper level of context (retrieval augmentation) for an LLM when generating a response

FINAL THOUGHTS

- AI-based conversion of unstructured content is here now; but a lot depends on how we use it
 - Despite the many “cons”, these can almost certainly be worked out with a dedicated AI/LLM for the task
- Definitely requires a human in the loop to check that the structured content is valid, both from a code and human-readable perspective
 - AI-based conversion is an assistive technology
- If structured content provides better/more correct answers for Chatbots, expect this to be a driving force—from a business perspective—to create and curate more structured content





Q/A