

An Illustrated Discussion of the Concept and Design of Pathagoras' Document Assembly System

Sorry about the dry sounding title, but there really is some 'exciting stuff' in here. It is important (especially if you have used other document assembly programs) to understand at least a little bit about *our* approach to building a document assembly system. It should make the rest of the pieces contained on other pages of the site fall into place a little easier.

A document assembly *system* is not just the engine that cobbles together documents from disparate parts. A document assembly system also must include:

- the means to create individual ("source") causes from which documents can later be assembled;
- the methods by which the various source clauses can be effectively recalled, displayed, selected and inserted into a final product;
- the techniques and tools for creating 'variables' (place holders) at strategic places throughout the source clauses;
- an efficient means to replace those variables with personal data after a document has been assembled; and
- an efficient means to store and then recall assembled documents for later editing.

Pathagoras contains all of these components. This overview touches on all.

Please note that the illustrations and examples which follow are quite simplistic. They are intended to convey concepts, not instruction. The omitted details, possibilities and optional approaches are explained within the program itself. Each module and overlay screen within Pathagoras contains many 'help' buttons, information screens and step-by-step instruction. The website (www.pathagoras.com) is also an excellent source of information on maximizing the benefits of Pathagoras.

If you have tried other document assembly programs, you will discover that Pathagoras works quite differently from what you have seen in the others. We invite you to read the discussion on the next page which describes the main 'differences' between Pathagoras and other document assembly programs you may have tried. Otherwise, you may wish to jump to page 3.

Differences on the document assembly side:

With most other document assembly programs, the order of document creation is: (1) the user completes a personal data form (or recalls an existing one) in order to create an unbreakable connection between the document being built and the client or customer. (2) The document is then assembled

Pathagoras' approach is the reverse. (1) The document is first assembled (or a complete form is called to the screen). (2) Client/customer information is applied last. Pathagoras simply rejects the idea that connecting a client to the document about to be built should be a mandatory first step. The link can wait.

Our customers have commented that they prefer Pathagoras' approach. It is more intuitive because it more closely mimics the flow they followed *before* trying a document assembly program. (Recall document, apply personal data.) And it allows them access to a broad range of pre-existing documents:

- a. Pathagoras' InstantDatabase (IDB) system is used to personalize documents (i.e., replace document variables with customer specific data). Pathagoras scans the document on the screen and identifies any text that appears in in between brackets as a 'variable.' (E.g., "[Client Name]"; "[Customer City]"; etc.) It displays those variables onto the IDB screen where you can apply new 'personal' client/customer data or recall an existing record. This process works with forms and clauses specifically prepared by you for use in your document assembly system. But it also works with any other document, including third party forms, where the variables are marked with [brackets]. Pathagoras automates those documents with no further editing.
- b. During the development phase, as you are building your clause libraries, it allows you to experiment 'with abandon.' You never have to link with a client, not even a 'dummy' client, in order to test your work.

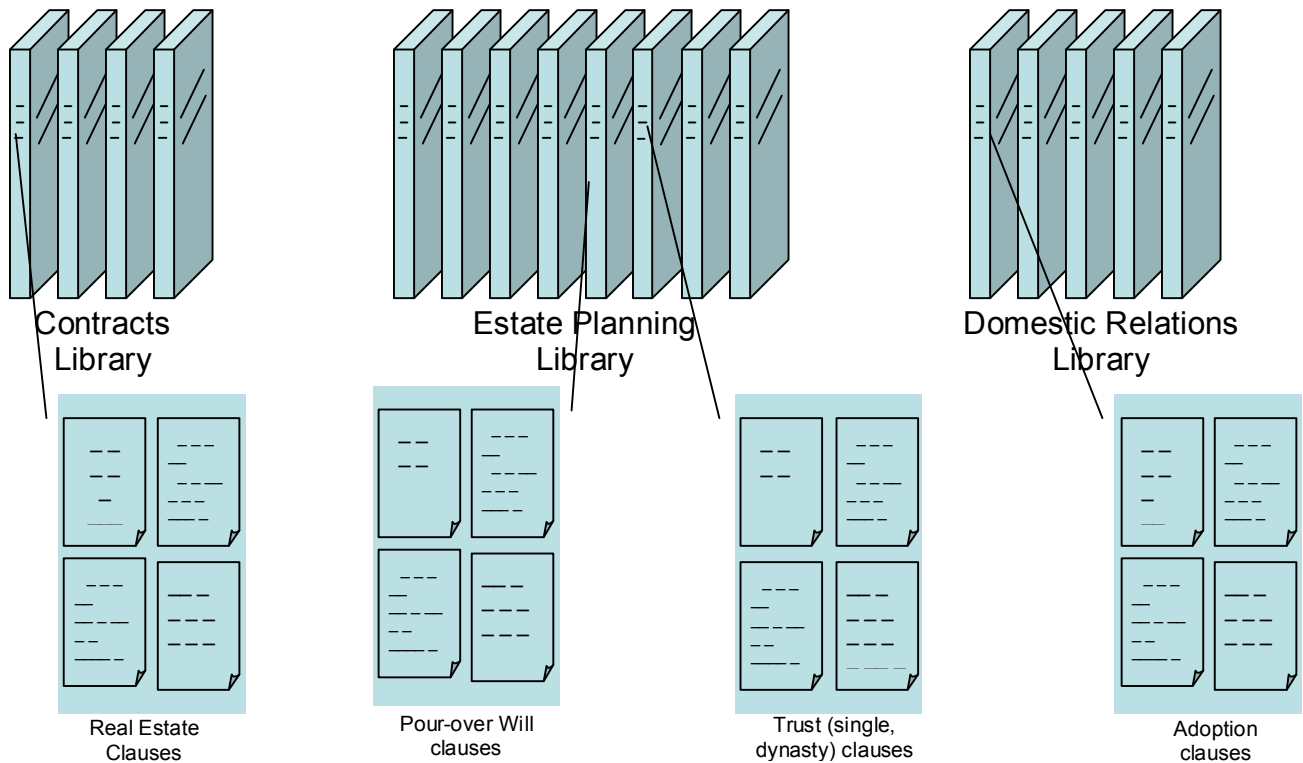
Differences on the clause and form creation side:

With most other document assembly programs, you initially have to create a large 'master' document (sometimes called a 'template'). This master contains every possible textual variation which will lead to a final product. When you want to add more options, you return to the master and code in the new text.

There is no 'master document' required. Pathagoras is a true 'paragraph assembler.' Pathagoras likes to build documents from collections of individual clauses. Your clauses. Clauses that are stored as Word documents, not as database code in another location by another program. Building from clauses stored as Word documents allows you a degree of flexibility and control (and intuitiveness) that is simply not provided by other approaches.

And most importantly, there is no coding, no hidden fields, no 'SmartTags' needed and that other programs demand. Personal data, quantities, gender and pronoun choices are all handled with plain text.

Libraries, Books & Clauses



Pathagoras Document Assembly follows a 'Libraries – Books – Clauses' metaphor. A **'library'** is a collection of books. A **'book'** is a collection of **'clauses'**.

A library can contain up to 10 books. Typically each book in a particular library will be on the same general topic. Pathagoras allows an unlimited number of libraries.

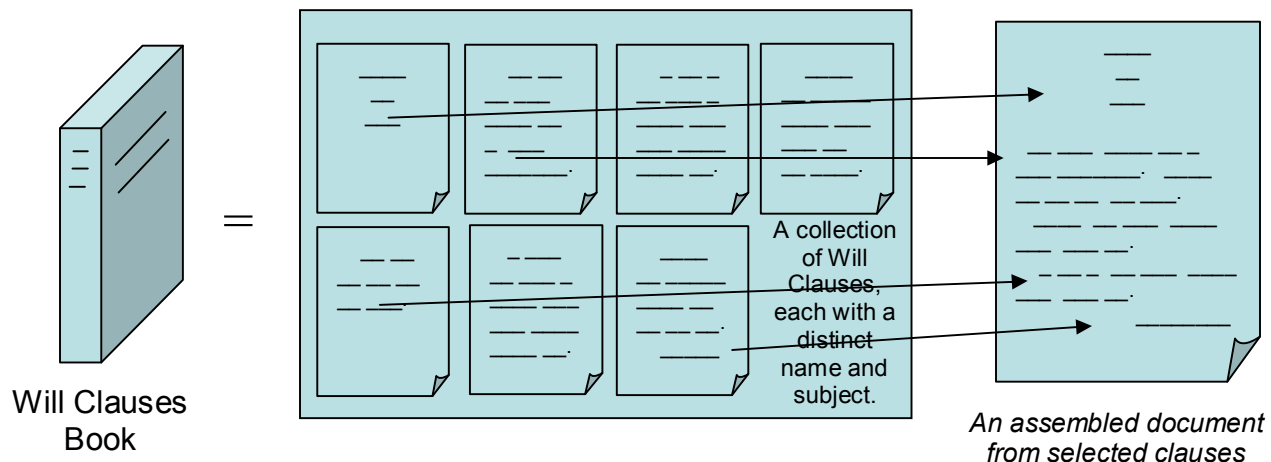
A **'book'** is typically standard Windows folder of documents. Nothing more. It contains the various **clauses** needed to create (assemble) a document in its designated subject area. A **clause** can be anything – words, sentences, paragraphs, pages, pictures, charts, etc. A clause can even be a complete document (avoiding the 'assembly' part of document assembly). Clauses can represent sequential sections of a document. Clauses can also be alternative versions of the same topic, allowing the user to create practically infinite variety in document creation.

So where do clauses come from? Typically they come from your existing documents. Simply take a 'favorite document' and break it up into its separate components, saving the components as individual documents in a folder. Add alternative clauses as appropriate to add potential variety to documents you will later assemble.

Once the book is created, document assembly can occur. To assemble a document, you would typically call up a library, select the book and then choose the clauses. Other ways that Pathagoras provides to move text from a book into a new document are discussed on the next page.

*A book can also be a 'glossary' of terms all stored within a single document. Each term in a glossary is separated from the others by bookmarks, which allow for quick retrieval. Glossaries are powerful tools, but are a more advanced feature. They are discussed in separate manuals.

Document Assembly via Books & Clauses



Pathagoras assembles documents from ‘books’ of clauses. A book¹ can comprise dozens or even hundred of clauses. Typically these clauses relate to a specific topic. For example, “Proposals”; “Wills”; “Real Estate”, etc.²

During document assembly, selected clauses are merely (but automatically, and very quickly) copied from the selected book and pasted into your document under construction. Pathagoras provides a variety of user tools to accomplish this routine:

(1) **Clause Selection Screen.** This is the ‘standard’ assembly method. Pathagoras displays the clauses in the chosen book onto a list. Select just the relevant clauses for the particular client’s or customer’s need. Press “<Next>” and you have an instant and complete document, ready to personalize.

(2) **Dropdown lists.** Pathagoras can display onto a dropdown list all (or a selected portion) of the term names in a particular book. The lists are always open in your Word menu area. Up to 10 such lists can be created, making all of the terms in 10 different books immediately available. It is just ‘point-and-click’ after that. Best used for insertion of single clauses (you know—“that one last clause”).

(3) **Manual insertion.** Type the clause’s name onto the screen and press <Alt-G>. This method works when the clauses have been named using the prefix/suffix naming style. Assuming the prefix and the book have been properly paired, Pathagoras will find the clause in the paired book (indicated by the prefix) and instantly insert the clause into your document. Best used for insertion of single clauses.

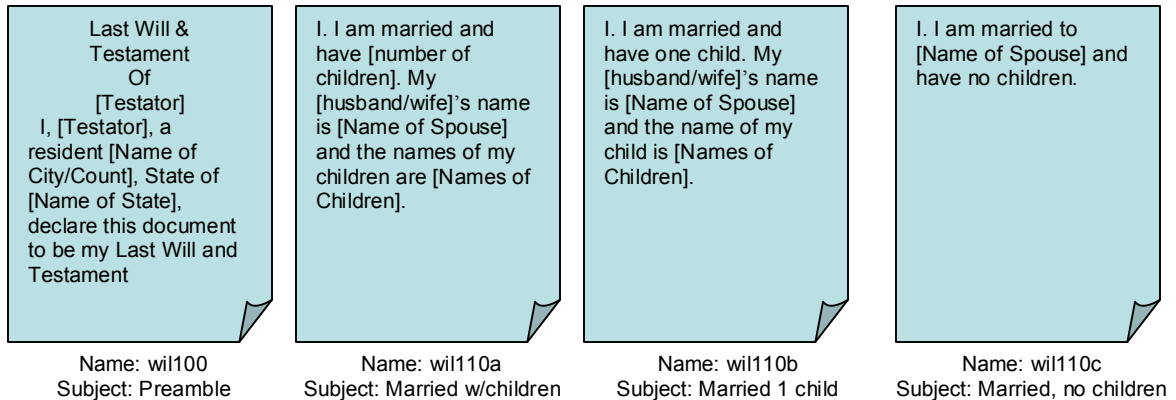
(4) **Clause sets.** A clause set is simple a Word document created by Pathagoras that allows a user to instantly assemble a document from pre-chosen clauses. Clauses sets can be stored in any book, each referencing different ‘packages’ of clauses for different situations.

(5) **Optional text.** These are sections of text built into a clause that will trigger an ‘include or exclude’ question for the user to answer. The question is presented at document assembly stage immediately after the document has been built. Optional text blocks are easily constructed. The user need only add the boundary markers ‘<<’ and ‘>>’ and the introductory word *Optional* (to make the entire block optional) or *Options* to allow the user to choose among a list of options, e.g., <<*Options*Next day delivery/3 day delivery/5 day delivery>>. Up to 5 options is possible within each set of <<*Options*>> brackets.

¹A book can either be a glossary (all clauses in a single document) or a folder (clauses saved as individual documents in the same folder). The discussion in this section presumes the latter

²In addition to topical books mentioned above, one or more ‘general glossaries’ can be created to hold non-subject specific data. For example, addresses, signature blocks, standard phrases (“To Whom It May Concern” or “Memo To:”), letterhead setups, etc., might be stored in a General Glossary.

The 'Anatomy' of Clauses



Pathagoras is a true 'paragraph assembler.'¹ It builds documents from 'the ground up' from clauses that are contained in 'books.' A book can contain dozens, even hundreds of clauses. Those above illustrate several important points about clauses and variables that are unique to Pathagoras:

(1) Note the simple and plain text nature of the clauses. As you will see when you use Pathagoras, 'plain text' doesn't mean unformatted text. Your text can be as rich and colorful as you wish. Here, plain text simply means 'no codes' or hidden fields.

(2) Note also the plain text nature of the *variables*. Variables are simply words within [brackets]. Even 'multiple choice' variables are incredibly simple to create. In the above example, [husband/wife] is a multiple choice variable. The slash makes the variable multiple choice. Like other variables, a multiple choice variable is constructed with plain text characters.

(3) Note the 'component nature' of 'paragraph assembly.' Pathagoras encourages users to create a variety of clauses and store them individually in the designated book. (Note also that wil110b and wil110c are merely variations of wil110a. Number, gender, condition, etc. variations can be provided for simply by adding of appropriately worded optional clauses.

(4) Note the prefix/suffix naming style (optional) along with the more descriptive 'Subject'. As you will see in actual use, this 'prefix/suffix' naming style adds tremendous power and flexibility. The prefix is stored in a separate table allowing the user an instant connection to the clauses folder. The numbers allow the user to predetermine the order that the clause will be presented in the clause selection screen.

(5) Note that the text is *your* text. It is not a pre-fabricated collection of clauses that you may not prefer to use (or pay money to update).

(6) A user can actually have a fully functional document assembly system with clauses as simple as those in the illustration. Start slow and small, and build from there.

¹This differs from many other programs. Most start with a large master template which contains all possible clauses and variations. The unnecessary parts of the template are pared away in response to answers to questions posed to the end user.

Instant Glossary

Last Will &
Testament
Of
John Q. Pathagoras

I, John Q. Pathagoras, a resident Athens, State of Georgia, declare this document to be my Last Will and Testament.

1. I am married and have three children. My wife's name is Patty Pathagoras and the names of my children are Plato Pathagoras, Athena Pathagoras and Hercules Pathagoras.
2. I direct that my hereinafter named Personal Representative shall pay all of my just debts from my estate.
3. I leave all of my worldly belongings to my wife, but if she shall not survive me, I leave such belongings to my children, in equal shares.
4. I appoint my wife to be my Personal Representative.

In witness whereof, I affix my signature.

John Pathagoras

Original document

(*)Last Will &
Testament
Of
[Testator]

(*)I, [Testator], a resident [Name of City], State of [Name of State], declare this document to be my Last Will and Testament.

1. (*)I am married to and have [number] children. My [husband/wife]'s name is [Name of Spouse] and the names of my children are [Names of Children].
2. (*)I direct that my hereinafter named Personal Representative shall pay all of my just debts from my estate.
3. (*)I leave all of my worldly belongings to my [husband/wife], but if [he/she] shall not survive me, I leave such belongings to my children, in equal shares.
4. (*)I appoint my [husband/wife] to be my Personal Representative.

(*)In witness whereof, I affix my signature.

[Testator]

Marked-up document

A 'first step' in building a document assembly system might be to select a favorite 'base' document and break it up into its component pieces. (This is repeated for all of your base documents until you have an entire system of libraries and books.) These pieces (and variations of each piece as may be needed to account for gender, number, condition, etc.) comprise the 'books' from which your documents can later be assembled.

Pathagoras provides easy to use tools by which to accomplish this document 'disassembly.' The example here illustrates how one might create a glossary using the '**Instant Glossary**' tool. (Other book creation tools exist. They are discussed on the website.)

Markup #1: Mark where each new clause is to begin. Any marker will do. We have used "(*)" in the example shown above.

Markup #2: Change 'real names' and other personal references to variables. A variable is simply a place holder for personal information. It can be any word(s) of your selection. Enclose variables within brackets for maximum flexibility. Multiple choices within a variable are easy to create, and are noted by a slash between the choices (e.g., "[husband/wife]"). You may provide up to 5 choices within a variable. Note that all components of a variable are 'plain text.' No codes or fields or Smart Tags are ever required in Pathagoras.

Once done, select Pathagoras' **Instant Glossary** tool from the Pathagoras menu. **Instant Glossary** It will determine the begin and end points of each clause (using your markers) and automatically create a fully functional glossary (collection of clauses). Complete instructions, of course, are provided within the **IG** module itself.

Personalizing the Assembled Document

(Replacing variables with personal values.)

Last Will &
Testament
Of
[Testator]

I, [Testator], a resident [Name of City], State of [Name of State], declare this document to be my Last Will and Testament.

- 1 I am married to and have no children. My [husband/wife]'s name is [Name of Spouse].
- 2 I direct that my hereinafter named Personal Representative shall pay all of my just debts from my estate.
- 3 I leave all of my worldly belongings to my [husband/wife], but if [he/she] shall not survive me, I leave such belongings to my [alternate beneficiaries], in equal shares.
- 4 I appoint my [husband/wife] to be my Personal Representative.

In witness whereof, I affix my signature.

[Testator]

Let's imagine that the Will glossary (the beginnings of which was discussed in other sections) was finally created. Then, a document similar to that at the left was assembled. (Note that the sample assembled document at left is not simply a duplicate of the original illustrated in earlier sections of this pamphlet. Clauses were added to the glossary so that we could create, in this example, a Will for a couple that does not have children.)

After the document is assembled, the variables await replacement with personal information. Pathagoras provides two distinct methods to personalize documents. The first is called "**Got Forms?**" When activated, **Got Forms?** will stop at each bracketed variable and ask for replacement text. Type the text in the space provided, and then click the <Replace> button. All identically named

variables will be replaced at the same time. Multiple choice variables (e.g., "[husband/wife]") can be completed by a click of a button generated on the **Got Forms?** screen. **Got Forms?** is activated with the key strokes <Alt-S>.

More powerful than **Got Forms?** (which is pretty powerful in its own right) is Pathagoras' **Instant Database** module. Like **Got Forms?**, **Instant Database** will identify each bracketed variable and provide space to type the personal information. It will also replace all identically named variables at the same time. But instead of replacing one variable at a time (as does **Got Forms?**) **Instant Database** processes all variables at one time.

Further (and more importantly), **Instant Database** allows you to save the associated "variable-personal data" combination as a recallable data record. Once saved, you can recall the data record as often as you need. Use it to complete other documents you want to create for the client or customer which contain the same variables. For example, in an estate planning practice, not only can you quickly personalize the Will as above, but so can you personalize the Trust, the Living Will, the Power of Attorney, the Deeds, all cover letters and letters of instruction, etc. needed to create an entire estate plan.

You can maintain an unlimited number of data records, and readily share them among all networked users.