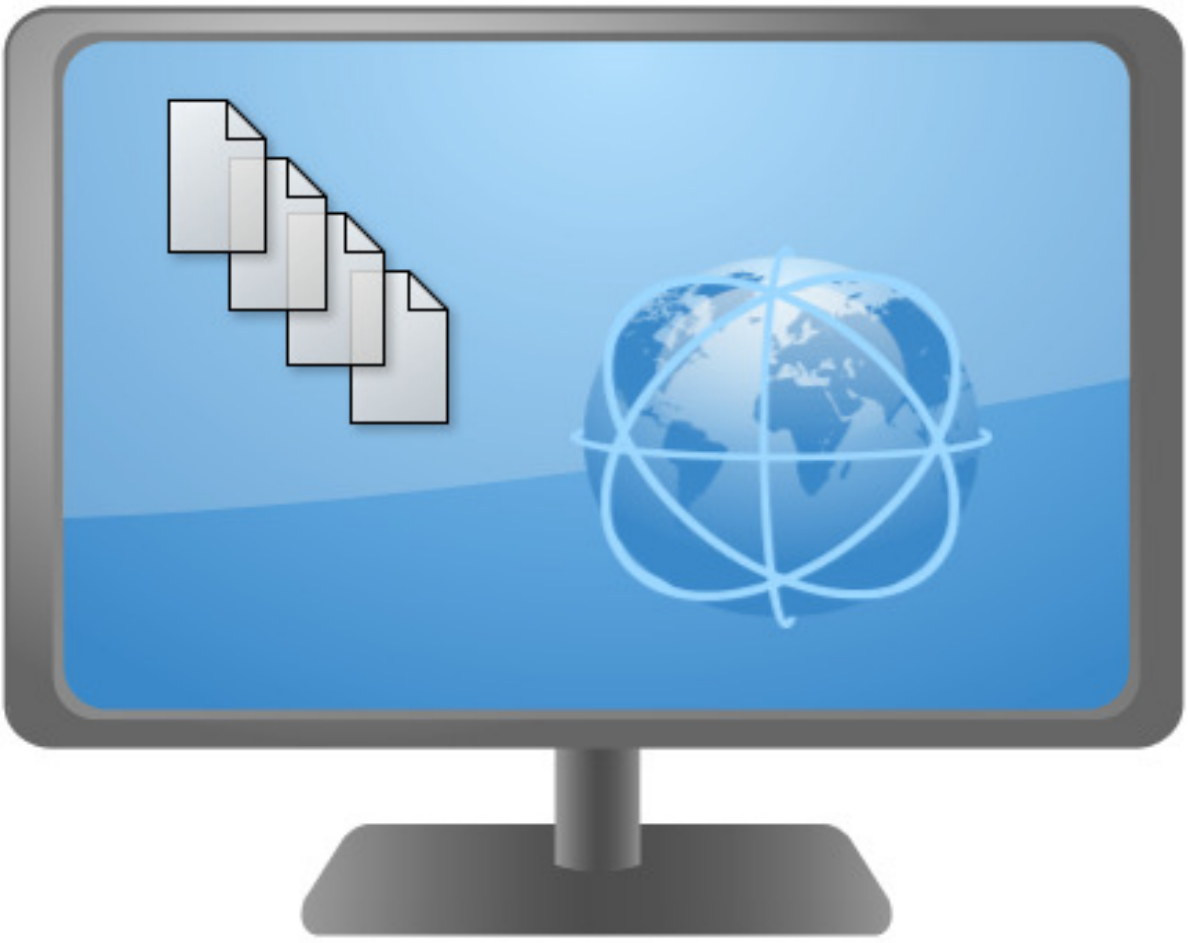
VideoRay DynaDoc



"DynaDoc is Awesome!"

Copyright Notice

Table of Contents

About this Document

Document Conventions

Several symbols are used throughout this documentation to add emphasis and to assist in relocating important information. The following table describes these symbols and their uses.

SYMBOL	DESCRIPTION
	The Danger icon is used to indicate there is a potential risk of personal injury or death. Extra care should be taken to understand the risks, and all personnel should exercise caution. It may also be appropriate to warn others in the immediate vicinity.
	The Caution icon is used to indicate there is a potential risk of damage to the equipment or surrounding property. Personnel should receive training in the appropriate procedures before attempting to operate or maintain the equipment.
	The Note icon is used to emphasize a specific detail or point of information.
	The Tip icon is used to highlight a suggestion.

Quality Commitment

VideoRay strives to design, manufacture, deliver and support the highest quality products and services, including this documentation. We have made every effort to ensure that this documentation is accurate and provides you with the most up-to-date information. However, there is no substitute for experience and/or training, especially with respect to the real purpose for which you plan to use this equipment. We encourage you to explore options beyond the scope of these materials to expand your knowledge and skills necessary to support your application. In addition to this documentation, VideoRay offers training and technical support and hosts a general user discussion forum and user image gallery.

We also realize that collectively, users of our products spend considerably more time operating our systems than we do ourselves. Users also encounter more diverse operating environments across an extremely broad range of applications. We highly value this vast experience base, and if you have any questions or suggestions, please feel free to contact us by any of the following methods.

If you find anything wrong with this documentation, or have suggestions for improvements, each page contains a "Help us improve this document" feedback link in the left margin (you must be connected to the Internet).

Address

VideoRay LLC
212 East High Street
Pottstown, PA 19464
USA

Email

info@videoray.com General Information and Sales
support@videoray.com Technical Support

Telephone

+1 610-458-3000 Office
+1 610-458-3010 Fax

The information contained herein is deemed accurate at the time of printing and is subject to change without notice.

Online Manual

The full version of this manual is available in HTML or PDF formats for viewing or download from VideoRay's website at: http://www.videoray.com/support/manuals.html.

How to Get Help

Help for your DynaDoc product is available through several channels.

All Hours Self-Service / Crowd-Source Tools

Operator's Manuals and Standard Operating Procedures	www.videoray.com/support/manuals.html
Software Downloads	www.videoray.com/support/downloads.html
Frequently Asked Questions	www.rovfaq.com
ROV User Forum	www.rovinfo.com

Global Support

Email	support@videoray.com
Phone	+1 610-458-3000 (*select option* 1)
Skype	videoray.support (*by appointment*)
Remote Sessions	www.videoray.com/support/remote-support.html (*by appointment*)

Regional Support

VideoRay Authorized Service Centers and Dealers	www.videoray.com/dealer.html

Training

Operator Training	www.videoray.com/learn-more/training.html
Advanced Maintenance Training	www.videoray.com/learn-more/advanced-maintenance-courses.html

Operational Strategies and Tactics Support

If you need help understanding how to apply your system to a specific project, contact VideoRay or you local VideoRay dealer. We can provide guidance or help you find a certified consultant.

About DynaDoc

DynaDoc is a semi-automated system that helps create multipage hierarchically structured documents in HTML format. It allows authors to focus on generating content, not page layout, linking pages or updating links when changes are made. It allows for the easy restructuring and reuse of content, and content blocks can be reused across several documents. Different versions or subsets of documents can be easily generated from the same base content using conditional pages and text within pages. Multiple documents can be easily cross-linked for seamless integration and consistent presentation styling. Styles can be changed and reapplied to create a completely different looking looks for the same document.

From the document viewer's perspective, DynaDoc provides an attractive, easy to navigate document. It is easy to point someone to a specific page by providing a URL link to that page. A "My_Notes" feature allows viewers to add notes on a page-by-page basis to local copies of the document.

DynaDoc is designed to complement automatic document generation systems and add the presentation layer. For example, DynaDoc is used to automatically create the presentation layer for several thousand page documents and maintain links between the pages that would otherwise be impractical to maintain manually.

This document was created using DynaDoc. It provides information about DynaDoc and instructions for using DynaDoc to create structured, feature rich HTML documents.

A complete list of DynaDoc's features is presented in the next section.

Why DynaDoc?

DynaDoc was created to solve a problem: How do you write and maintain good looking structured HTML documents?

Benefits of using HTML:

- All content and display is based on open formats
- Deliver files from a server or local file system
- Supported by every browser without helper apps
- Viewable on many device types
- Link directly to a page or page section

However, when creating large multiple page documents, there are several challenges:

- No automated table of contents or index.
- Not easy to change page layout significantly (CSS notwithstanding, but removing a footer, adding a column or making other structural changes would require editing all of the HTML files).
- As the number of pages grows, managing document navigation links manually becomes impractical if pages are added, dropped or renamed.
- As the number of pages grows, managing a sequential menu or table of contents manually becomes impractical.

DynaDoc solves these challenges, and more...

DynaDoc Features

Document Creation Features

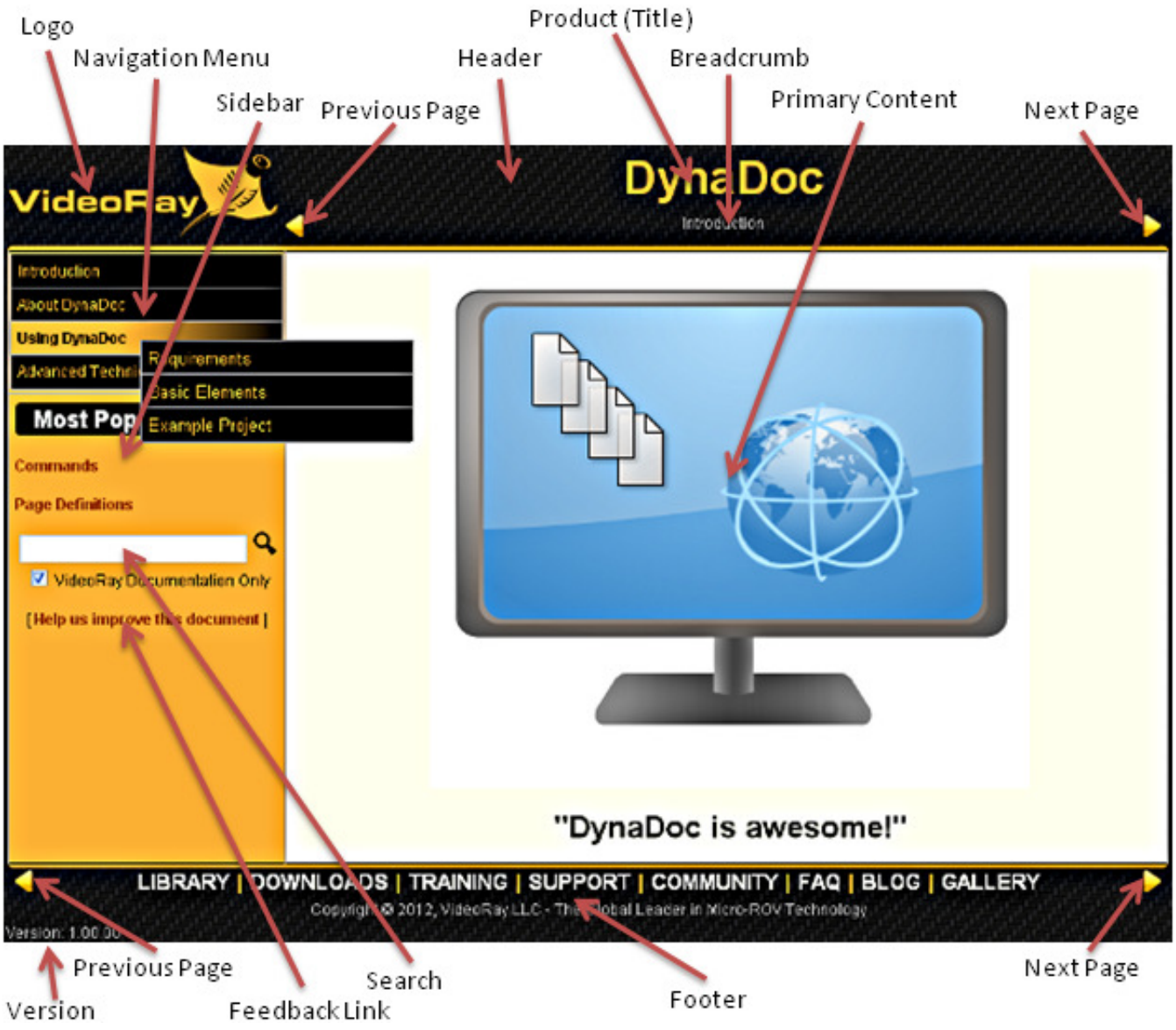
- Automatic creation of pages using a template for page layout and content blocks (currently 1 template for the entire document, but this could be extended).
- Automatic creation of document navigation menu and links
- Automatic creation of previous page, next page and "up to higher levels within the hierarchy" links
- Automatic creation of a table of contents
- Automatic creation of a linked index to specified keywords
- Image support in menu and Table of Contents
- CSS / page template based layout, separated from content blocks
- Multiple content blocks per page (currently 2 blocks per page, but this can be extended. Each block is optional)
- Content is created using simple, modular, reusable content blocks
- Self-contained output that can be viewed from a server or from the local file system
- Simultaneous creation of single html page for PDF generation
- Conditional page output for viewing version only, print version only or both (default)
- Conditional text output for viewing version only, print version only or both (default)
- Conditional pages (30 unique identifiers, multiple can be used simultaneously)
- Conditional text (9 unique identifiers, multiple can be used simultaneously)
- JavaScript can be used for dynamic user interaction on a page by page basis

Document Viewing Features

- Easy menu-based navigation
- Linked Table of Contents
- Linked Glossary / Index
- Use external links directly to specific pages
- Language translation support (powered in real time by Google Translate)
- Google Search integration
- My_Notes feature allows users to make notes on a page by page basis (on local copies)
- Feedback link on a page by page basis

DynaDoc Results Example

The following image shows the typical page elements of a DynaDoc generated page.



FAQ (Frequently Asked Questions)

1. **What do I need to know to get started with DynaDoc?**

   Basically, you only need to know how to write HTML pages. See the Requirements section for more information.

2. **How big of a documentation project can DynaDoc handle?**

   DynaDoc can be used with documents consisting of over 1,000 HTML pages.

3. **Why use DynaDoc instead of Word (PDF, etc.)?**

   DynaDoc has many features and benefits. See the Features and other sections for more information.

4. **How do you print multipage HTML documents easily?**

   DynaDoc generates a consolidated print file that contains all of the HTML pages in one file for easy printing.

Using DynaDoc

DynaDoc is PHP program that is executed by entering its URL in a browser. It reads "raw" document source files and generates the final document as set of linked HTML pages. Documents to be created using DynaDoc are written as "content blocks" in HTML (one file per page). DynaDoc also requires a page layout template and text "script" to define the sequence of the pages.

Document Creation Process

Creating a document requires some set up steps.

1. Create a folder structure for your project.
2. Create page templates defining the layout of the view and print pages. The templates needs to include tags for where content will be placed.
3. Create any desired CSS. External CSS files are recommended.
4. Create a script. The script is like the outline of the document. For each page of output, there needs to be one line in the script. The script also contains some commands that define various parameters of the document.
5. Create content blocks. Each content block is an HTML file. HTML should be limited to simple elements, such as headings, paragraph control, images, links, etc. Page layout is handled by the template.
6. Create any JavaScript (general or page specific).

Once the set up is complete, the only remaining step is to run DynaDoc. This is accomplished by providing the URL to DynaDoc in a browser. DynaDoc requires the Project Name.

Example:

- **http://localhost/dynadoc?proj=**<*projectName*>

Other arguments can be provided. The list of all arguments include:

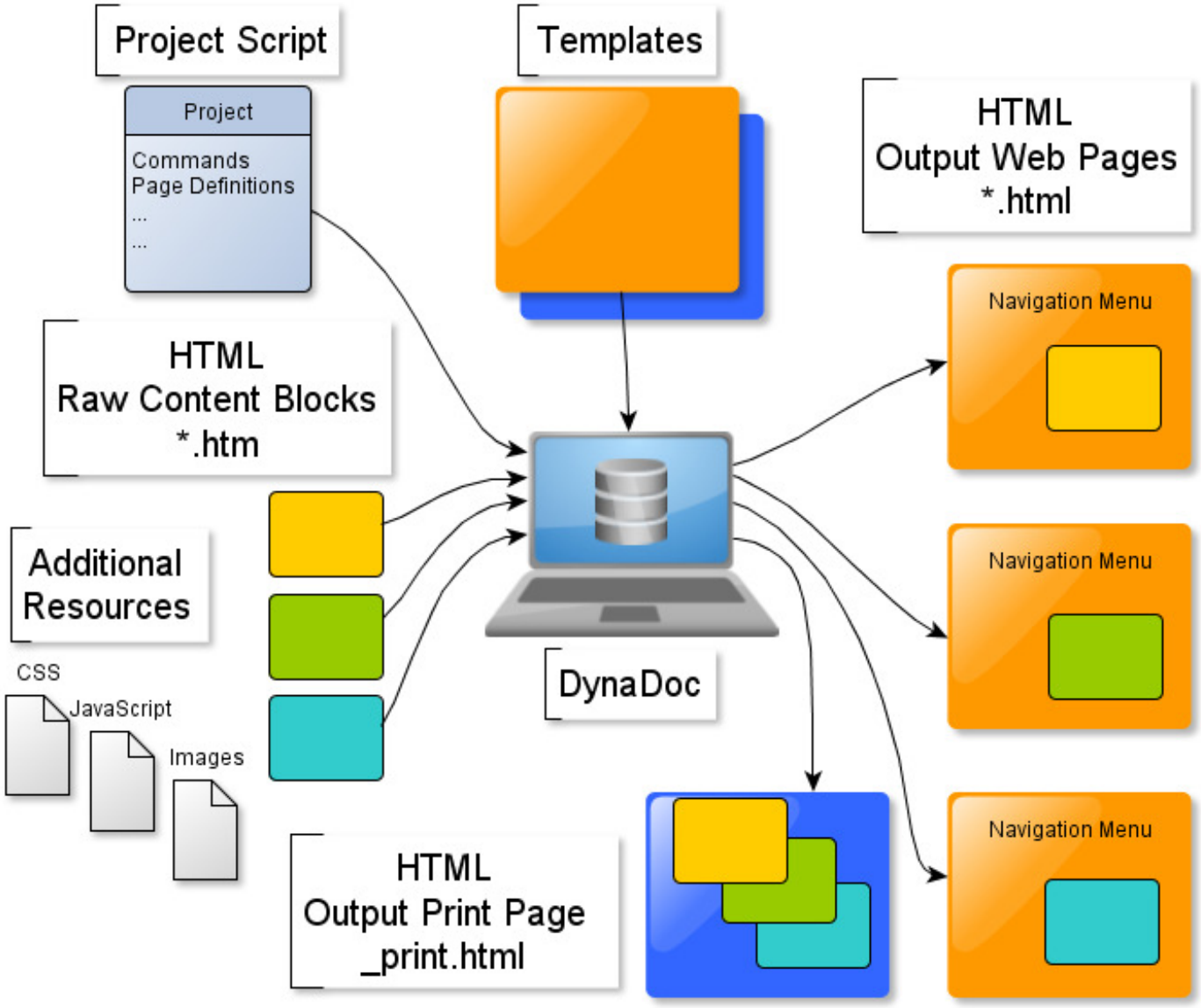
Argument	Purpose
name	This is for display purposes only.
proj	The name of the project script (do not include .txt).
projDir	The name of the project folder if different from the project script. projDir requires the trailing slash, but one will be appended if it is not included..
outDir	The output folder if not the same name as the project folder. outDir requires the trailing slash, but one will be appended if it is not included. The default ouput path is out/, but this can be changed.
rootDir	DynaDoc expects the project folder to be in the same folder as dynadoc.php. If the project is in a lower level folder, set rootDir to the path to the project folder. rootDir requires the trailing slash, but one will be appended if it is not included. rootDir defaults to no value (blank).
docList	If docList is set to any value other than an empty string, "", DynaDoc will process a list of script files. The DynaDoc list file is a text file of the names of the scripts with one script file name per line (do not include .txt). The list file's name is of the form *proj*.lst.
editMode	If editMode is set, DynaDoc will replace the <!–##EDIT_LINK##–> tag with a link to an edit script. The <!--##EDIT_LINK##–> is usually set in the view template and can be used to create a link to allow editing of the htm file using ckeditor or other editor.

When working on multiple projects, DynaDoc is also set up to work with project lists as input. See the Advanced Techniques section.

DynaDoc will do the following:

1. Using the script, a linked menu is created in HTML format.
2. For each page definition in the script, the project name, version, menu, content blocks and JavaScript (optional) will be merged with the template to create a corresponding output page.

A graphic representation of this process is provided below:



DynaDoc will actually create two sets of output. One for viewing, and two print versions. Typically, the viewing version will be complete (it is easy to navigate and disk space is cheap), and the print version will be a subset (think green). The print versions will consolidate all pages to single files for printing purposes. One print version will include only those pages specified to be included in the print version. The second print version will include all pages from the viewing version.

Document Viewing Process

The final document will be contained in a folder structure defined by the user. The root folder of this set can be posted on a domain or a local file system. Any HTML file within the output can be opened in a browser for viewing, either by specifying the URL, or dragging and dropping the file on a browser window. Typically, the first page of the document will be named index.html for ease of use.

Advanced Features

There are other advanced features and capabilities of DynaDoc that are described later in this document.

Requirements

DynaDoc is designed for use in a development environment and requires a web server with PHP enabled. It can be used on an externally hosted server, but this is not conducive to production level creativity and editing cycles. Apache Friends provides a free XAMPP package, which includes everything required to install a local web server with PHP enabled.

Application Requirements

In order to use DynaDoc, you will need the following:

- Web server with PHP enabled.
- Access to upload or copy files to a folder in the html path of the web server.
- The DynaDoc PHP program suite.
- View page and print page HTML templates.
- style_menu.css - the document uses CSS to display the navigation menu. This file controls the display of the menu and is required. The use of additional CSS is optional.

Experience Requirements

DynaDoc uses commands, scripts and content blocks. This document will provide the information needed to learn how to use these elements and the procedures required to create documents using DynaDoc.

Content blocks are HTML formatted files, so knowledge of HTML is required. However, content blocks can be created using relatively simple HTML, so an advanced knowledge of HTML is generally not required. If you want to modify the layout template, you may need advanced HTML and CSS knowledge.

PHP knowledge is not required, but may be helpful, and is required if you want to customize DynaDoc.

Architecture

Dynadoc consists of a series of PHP programs and configuration files that are used to generate and administer projects. DynaDoc programs are designed to be used with arguments, but for easy execution, html files can be created using links or buttons and dynadoc_manage.php is designed to call some of the other programs.

The DynaDoc root folder of the web server should contain these files and each project requires its own folder in the same root.

Document projects can be arranged in categories to make management and operations easier.

Programs include:

1. dynadoc.php - The primary program for generating documents
2. dynadoc_create.php - A utility to create new projects
3. dynadoc_edit.php - A utility to allow editing HTM files in the browser (any editor may be used)
4. dynadoc_form.php - A utility to run dynadoc using a form for input
5. dynadoc_glossary.php - A utility to generate a page with a linked glossary/index
6. dynadoc_images.php - A utility that helps manage document images
7. dynadoc_manage.php - A utility to facilitate launching DynaDoc programs
8. dynadoc_orphan.php - A utility to find content blocks (HTM files) that are no longer used in the project
9. dynadoc_projects.php - A utility to create a list of existing projects
10. dynadoc_publish.php - A utility to prepare a document file set for publishing
11. dynadoc_zip.php - A utility to create a self extracting zip file of the project
12. sop_checklist.php - A utility to create a checklist from a Standard Operating Procedures manual

Configuration files include:

1. dynadoc_create.lst - a list of categories
2. dynadoc_*catName*.lst - a list of projects in the *catName* category
3. dynadoc_publish.lst - a list of standard folders and files to be published

If starting from scratch, you will need to create the category list file (dynadoc_create.lst) and a dynadoc_*catName*.lst file for each category. An example category list file is shown below:

```
categoryName|displayName|
_rov|ROVs|
```

The first line is a header that describes the record format. The second and subsequent lines include a category mnemonic and display name. In this example, **_rov** is the *catName* and **ROVs** is the display name. This file must be created and updated manually to add categories. At least one category must exist, and the trailing "|" is required.

An example category project list file (dyandoc_rov.lst) is shown below:

```
Name|Project|Source Dir|Out Dir|List?|
```

The first line is a header that describes the record format. Additional lines are automatically created by dynadoc_create.php.

- **Name** is the formal name of the project.
- **Project** is the name of the project's script.
- **Source Dir** is the name of the folder for the project.
- **Out Dir** is the name of the folder for the output. If left blank, it will default to the **Source Dir**. If defined, it can be used to create separate outputs from the same source.
- **List?** is either blank or any character. If a character is used, DynaDoc will process a list of documents. Each project script name is placed on a separate line in the list file. The first line of the list file should be blank.
- The trailing "|" is required.

dynadoc_manage.php is the primary access point. The easiest way to use dynadoc_manage.php is to create a link to it in an HTML file such as index.html. An example line in an index.html file is shown below:

```
<a href="dynadoc_manage.php?category=_rov">ROVs</a>
```

Additional lines can be created for each category. This file must be created and updated manually.

Additional information about working with document projects can be found in the Document Creation manual.

Basic Elements

The basic elements of a DynaDoc project include:

Element	Description
Folder Structure	The Folder Structure helps manage the various elements and resources of the project. Typically, you will have a folder structure for the source and a separate folder structure for the compiled documents.
Templates	The Templates define the page layout. There is one template for the view version and one template for the print version. The template includes tags to control the placement of various page elements like the document name, version and content blocks.
CSS	External CSS files are recommended. styles_menu.css is required for controlling the display of the navigation menu.
Content blocks	A Content Block is an HTML file that contains the content for each page. Typically, there is primary content and an optional side bar content. The only styles and layout in these files should be to define headings, paragraphs, images, links, etc. If displayed directly in a browser, these pages will look "raw" and are referred to as the "raw" HTML.
Resources	Resources include images, videos and other page elements or attachments.
Script	The Script is a text file that contains the instructions to compile the document. It includes commands and page definition lines.

Folder Structure

Creating an organized folder structure is critical to establishing an efficient and productive environment. Typically, there will be a folder set for each document source, and a another folder set for each output.

An example source folder set is illustrated here:

sourcePath/project_name
- css
- glossary
- htm
- images
- javascript
- My_Notes
- pdf
- videos

An example output folder set is illustrated here:

outputPath/project_name
- css
- html
- images
- javascript
- My_Notes
- pdf
- videos

Items in these folders should be self evident based on the names of the folders.

DynaDoc convention is to name source files ".htm," and output files ".html." The output folder structure would be similar to the above, but the "htm" folder is named "html."

The DynaDoc script would typically be stored in the project_name folder.

Templates

There are two templates - one for the viewing version and one for the print versions. Each template is an HMTL page that embodies the structure and layout of the document, but does not have content. The templates contain tags that are placeholders and are replaced with content or other information during document creation. The use and placement of tags is entirely up to the author and how the results will look depends upon where the tag is placed in the template and any CSS applied.

The following tags are available:

Tag	Purpose
<!--##MENU##-->	Navigation Menu
<--##TITLE##-->	Page title, shows up in the browser tab of the page being viewed
<--##PRODUCT##-->	Document Name, typically displayed as a heading at the top of each page
<--##VERSION##-->	Document version, typically displayed in the footer of each page
<--##PAGE_URL##-->	Page URL, the URL of the current page
<--##CONTENT_1##-->	Primary content block, usually the main focus of each page
<--##CONTENT_2##-->	Secondary content block, usually used as sidebar text
<--##JAVASCRIPT##-->	Allows page based JavaScript
<--##PREVIOUS##-->	Previous Page button
<--##NEXT##-->	Next Page button
<--##BREADCRUMB_STRING##-->	Breadcrumb that lists the document path to the page included links to higher levels within the path to the current page
<--##EDIT_LINK##-->	Can be replaced with a link on the html page to allow editing of the htm file using ckeditor or other web-based editor
<--##LOCAL_CSS##-->	Allows document-based CSS
<--##CSS_PATH##-->	Allows document-based path to CSS files
<--##HTML_PATH##-->	Allows document-based path to HTML files
<--##IMG_PATH##-->	Allows document-based path to images
<--##JS_PATH##-->	Allows document-based path to JavaScript
<--##VID_PATH##-->	Allows document-based path to videos

CSS

CSS can be used to apply layout and styles to the document. External CSS files are recommended. Typically, there will be a common CSS base used across all document projects, but custom CSS can be applied on a per project basis.

Content Blocks

Content Blocks contain the creative content of each document page. They are HTML files, but they do not include page level layout, or any of the document's structural elements like the title, menu, header, footer or search form. These elements are supplied by the template or DynaDoc. DynaDoc was designed so that the effort of creating documents is focused primarily on the creation of these content blocks, and the order in which they will be displayed. There are two content blocks that can be used on each page. One is used for the primary content of the page, and the second is used for a sidebar.

The primary content block for this page up to this point is shown below.

```
<h1>Content Blocks</h1>

<p>Content Blocks contain the creative content of each document page. They are HTML files, but they do not include page level layout, or any of the document structural elements like the menu, header or footer. These elements are supplied by the template or DynaDoc. DynaDoc was designed so that the effort of creating documents is focused primarily on the creation of these content blocks, and the order in which they will be displayed. There are two content blocks. One is used for the primary content of the page, and the second is used for a sidebar.<p>

<p>The content block for this page up to this point is shown below.<p>
```

Notice that the contents of the content block are quite simple, yet the page you see displayed has many more elements and features. That is the beauty and power of DynaDoc - to create this entire page, only the above block of HTML had to be written.

Other key points:

- The layout and style of this page can be easily changed by applying a different template.
- The document can be easily restructured and the menu would be regenerated by DynaDoc, so menu and link management is not required.
- Conditional pages and text can be used to create unique versions of this document from the same source.

Content Block Tags

The primary and secondary content blocks allow tags for conditional text and to include other files. The following tag sets are allowed:

Content Block Tag Set	Description
<!–v--> <!–/v-->	Enclosed text will only be included in view version.
<!–p--> <!–/p–>	Enclosed text will only be included in print version.
<!–#--> <!–/#–>	Enclosed text will only be included if the VALID_TEXT command includes #, where # is 1 - 9.
<!–i--> <!–/i-->	Include another content file in place of this tag set. The file name should be included between the tags, and may include a path. Multiple include tags may be used in one file, and include tags can be used recursively - any include tags in the included file will be processed.

Sample Sidebar

This is a sample sidebar. The content of this sidebar is an HTML file. Whether a sidebar appears on a page or not is defined in the script.

Resources

Resources include JavaScript, images, and other page elements that are not part of the template or content blocks. The folder structure helps manage these resources.

Script

The script defines the how the document is to be compiled. It includes various document parameters (including the definition of some of the tags used in the templates) and the sequence of the document's content blocks. The script can include commands, page definitions, and comments. More information about these script elements is included in the following sections.

 Each line of the script must be complete (no line breaks), and each line can contain only one script element. Comments and blank lines in the script are acceptable and can be helpful in writing scripts that are easy to understand and manage.

Commands

Command Syntax:

!COMMAND,Parameters

Where:

Command Element	Description
!	Command identifier
COMMAND	The command name
Parameter(s)	Parameter(s) will depend upon the command

Commands for use in script (commands in **bold** are required, all other are optional)

Command	Parameter(s)	Description
BASE_URL (Reserved)	<URL>	Base URL of planned destination
COPY (Reserved)	<filenames and/or foldernames>	List of files (css, etc.) to copy to output folder, folders must end with a / (like images/) must be separated by commas
CSS_PATH	<pathname>	CSS path
HTML_PATH	<pathname>	HTML files include Path
JAVASCRIPT_PATH	<pathname>	JavsScript files include Path
IMG_PATH	<pathname>	Image files include Path
INCLUDE	<filename,level>	Include an external script, and the base level for pages. Can include a path and be used recursively. The INCLUDE command cannot be on the first line of script - if you want the script to start with INCLUDE, add a blank line before the INCLUDE line.
LOCAL_CSS	<pathname>	Project specific CSS file
MENU_WIDTH	<number>	Menu width in pixels
OUT_DIR	<foldername>	Output folder (include trailing /)
OUT_PATH	<pathname>	Output path (include trailing /)
PRINT_NAME	<filename>	Name of print file
PRINT_VIEW_NAME	<filename>	Name of print file for view pages
PAGE_BREAK_LEVEL	<level number>	Page breaks will be added to all pages at this level and all higher levels in the hierarchy. Defaults to 1, the highest level. If set to 3, all pages on levels 1, 2 and 3 will print on a new page. To suppress all page breaks, set PAGE_BREAK_LEVEL to 0. PAGE_BREAK_LEVEL can be overridden on a page by page basis.
PRODUCT	Document Name	Document Name
TEMPLATE_COVER	<filename>, <filename>	Cover Page Template, and cover page content
TEMPLATE_PRINT	<filename>	Print Page Template
TEMPLATE_VIEW	<filename>	View Page Template
TOCMENU_IMAGES	<blank or 0>	Add images to menu, default = yes, set to 0 to disable
TOCPRINT_IMAGES	<blank or 1>	Add images to print TOC, default = no, set to 1 to enable
TOCVIEW_IMAGES	<blank or 0>	Add images to view TOC, default = yes, set to 0 to disable
VALID_PAGES	<string of characters>	List of pages that are valid, defaults to vpxq$@is
VALID_TEXT	<string of numbers>	Used to control which conditional text is included, numbers range from 0 - 9, if a number is included in this list, the corresponding conditional text will be displayed
VERSION	Document Version	Document Version
VID_PATH	<pathname>	Video files include Path

Page Definition Lines

The page definition lines include references to the source files and define various aspects of the output.

The source files include:

Input	Description
Primary Content Block	The primary content block is inserted into the template based on the location of the <--##CONTENT_1##--> tag and is merged into one or more of the above files.
Secondary Content Block	The secondary content block is inserted into the template based on the location of the <--##CONTENT_2##--> tag and is merged into one or more of the above files. This input is optional.
JavaScript	A reference to a JavaScript file is inserted into the template based on the location of the <–##JAVASCRIPT##--> tag and is merged into one or more of the above files. This input is optional.
Image	An image can be included in the menu and Table of Contents. This input is optional.

Typically, the following output will be generated:

Output	Description
View file	A separate HTML file for each page, for viewing.
Print file	A single HTML file including every page, for printing.
View_print file	A single HTML file including only the pages defined to be viewed, for printing.
Navigation Menu	The navigation menu is inserted into the template based on the location of the <!–##MENU##–> tag and is merged into one or more of the above files.

There are numerous control techniques that can be applied within the page definition lines to define exactly what output is created. The details are provided below.

Page Definition Line Syntax:

Qualifier,Page_Title,Output,Primary_Content,JavaScript_Block,Secondary_Content,Menu_TOC_Image*

Where (items in bold are required, all other items can be omitted):

Page Definition Element	Description
*	Menu level. Use one * for the top level, two *s (**) for the next level, and so on. Levels can only increase one at a time from one line to the next (1 to 2, 2 to 3, etc.), but can decrease any amount (3 to 2, 3 to 1, 4 to 2, etc.)
Qualifier	Qualifiers define special processing and are included before the menu level • v == include in view only • p == include in print only • x == do not include in view • q == do not include in print • $ == generate output, but do not include in menu or print • @ == force a page break. Must come before * and cannot be used with i or s • ? == conditional page, where ? is a single digit. ? must come before v, p, x or q and cannot be used with i or s • i == special conditional page for instructor note pages, not included in menu. Must come before v, p, x or q. • s == special conditional page for student note pages, not included in menu. Must come before v, p, x or q. Note - There is no comma required between the qualifier and the menu level.
Page_Title	The page title is used as the text for the menu and the title of the page (displayed on the browser tab). Typically, this will also be the page heading in the primary content block. The menu may have limited space, so the page title might have to be shortened or abbreviated, but the page heading can be as long as you want.
Output	The file name of the output file. Output is generated by combining the content blocks and other

	parameters with the template. Typically, Output is omitted and the output name is automatically generated based on the name of the Primary Content Block. If the file extension of the Primary Content Block is ".htm" an "l" will be appended, so the extension of the output file will become ".html" If Output is included, it is used as the name of the output file.  "NUL" is special case that does not generate output. This can be used to create menu and table of contents entries and links to a file that already exists (such as an HTML file created earlier in the script or a PDF or other file type not created by DynaDoc).

Primary_Content	The file name of the primary content block - usually an HTML file using ".htm". If there is no path included in the name of the primary content block, the path is assumed to be the local "htm" folder. Otherwise the path is used to locate the file. The following actions take place for each page definition line:
	
	• A menu entry is created (unless the "$" qualifier is used)
	• The contents of the Primary_Content file are inserted into the template replacing the <!–CONTENT1--> tag
	• Based on the information below, output may or may not be created:
	  ◦ An output file is created for the view version (unless the "x" or "p" qualifier is used)
	  ◦ Output is added to the print file for the printed version (unless the "v" or "q" qualifier is used)
	  ◦ Output is added to the view_print file for the printed version (unless the "x" or "p" qualifier is used)
	  ◦ No output is created if the Output name is "NUL"
	
	  "NUL" as the Primary_Content is special case that creates an entry in the menu and Table of Contents, but does not link these entries to any page. No output file is generated.

JavaScript_Block	The file name of a JavaScript file. A reference is created to this file and is inserted into the template at the <!–JAVASCRIPT--> tag.
Secondary_Content	The file name of the secondary content block. Usually an HTML file using ".htm". The contents of the file are inserted into the template at the <!--CONTENT2–> tag.
Menu_TOC_Image	The file name of an image to be used in the menu and Table of Contents for this page.

 If an optional item is omitted, its place must be preserved by using commas. Example page definition line with elements omitted:

**,title,,page.htm,,,*

Comments

Comment Syntax:

#This is a comment

Where:

is the comment identifier and anything that follows is the comment.

Example Project

The source files for this document are available for examination and practice.

- DynaDoc documentation source (zip file)

The script and template files included on the following pages are included dynamically using <!--i--> <!--/i--> tags instead of copying or cutting and pasting them. This will ensure that at the time the document is generated, these pages will represent the latest versions of these files.

Example Script

The script for this document is displayed below:

```
#Document Commands
!PRODUCT,DynaDoc
!VERSION,1.00.00
!DOC_TYPE,Operator's Manual
!OUT_DIR,dynadoc/
!VALID_PAGES,
!VALID_TEXT,
!TEMPLATE_COVER,required/htm/template_index_print.htm,index_print.htm
!PAGE_BREAK_LEVEL,6

#Documentation Generation
!INCLUDE,required/intro_.txt,1

*,About DynaDoc,,about.htm,,,
**,Why Dynadoc?,,overview.htm,,,
**,Features,,features.htm,,,
**,DynaDoc Results Example,,results_example.htm,,,
**,FAQ,,intro_faq.htm,,,
*,Using DynaDoc,,using_dynadoc.htm,,,
**,Requirements,,requirements.htm,,,
**,Architecture,,architecture.htm,,,
**,Basic Elements,,elements.htm,,,
***,Folder Structure,,folder_structure.htm,,,
***,Templates,,templates.htm,,,
***,CSS,,css.htm,,,
***,Content Blocks,,content_blocks.htm,,content_blocks_sb.htm,
***,Resources,,resources.htm,,,
***,Script,,script.htm,,,
****,Commands,,commands.htm,,,
****,Page Definitions,,page_definitions.htm,,,
****,Comments,,comments.htm,,,
**,Example Project,,project.htm,,,
***,Example Script,,example_script.htm,,,
***,Example View Template,,example_template_view.htm,,,
***,Example Print Template, Cover,,example_template_print_1.htm,,,
***,Example Print Template, Page,,example_template_print_2.htm,,,

*,Advanced Techniques,,techniques.htm,,,
**,Additional Notes,,additional_notes.htm,,,
```

Example View Template

The view template for this document is displayed below:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
        <meta http-equiv="content-type" content="text/html; charset=utf-
8" />
        <title>DynaDoc Example View Template</title>
        <meta name="keywords" content="" />
        <meta name="description" content="" />
        <link rel="stylesheet" href="../../required/css/style_tags.css"
type="text/css" media="screen, projection" />
        <link rel="stylesheet"
href="../../required/css/style_classes.css" type="text/css"
media="screen, projection" />
        <link rel="stylesheet" href="../../required/css/style_ids.css"
type="text/css" media="screen, projection" />
        <link rel="stylesheet" href="../../required/css/style_menu.css"
type="text/css" />
        <!--##LOCAL_CSS##-->
<script type="text/javascript"
src="../../required/javascript/preview_checks.js"></script>
<script type="text/javascript"
src="../../required/javascript/preview_config.js"></script>
<script type="text/javascript"
src="../../required/javascript/preview.js"></script>
<script type="text/javascript"
src="../../required/javascript/show_hide.js"></script>
<!--##JAVASCRIPT##-->

</head>
```

Example Print Template, Cover Page

The print template for this document is displayed below:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
        <meta http-equiv="content-type" content="text/html; charset=utf-8"
/>
        <title>DynaDoc</title>
        <meta name="keywords" content="" />
        <meta name="description" content="" />

        <link rel="stylesheet" href="../../required/css/style_print.css"
type="text/css" />

<!--##JAVASCRIPT##-->

</head>

<body>

<div id="page">
  <div id="content">
<!--##CONTENT_1##-->
    </div>
  <div id="content">
<!--##CONTENT_2##-->
    </div>
  </div>
```

Example Print Template, Content Pages

The print template for this document is displayed below:

```
  <hr class="topLine">
  <table class="transparent">
    <tr>
      <td class="imageCellBorderless">
        <div id="content">
<!--##CONTENT_1##-->
          </div>
        <div id="content">
<!--##CONTENT_2##-->
        </td>
      </tr>
    </table>
```

Advanced Techniques

DynaDoc has some sophisticated capabilities that can be used to make document creation and managing complex documents easy.

Print Version Page Breaks

By default, page breaks are added to each top level page (based on menu hierarchy). You can change the level at which page breaks are applied by using the PAGE_BREAK_LEVEL command. You can also force a page break for any page.

Restructuring Documents

Documents can be easily restructured by changing the sequence of the lines in the script. It is easy to move pages, or split or combine pages, and you don't have to worry about updating the navigation links.

Using Conditional Pages and Text

Conditional pages and text allow one set of source files to generate several to many different outputs. For example, links and JavaScript have no value in a printed document. Conditional text can be used to eliminate this superfluous information from the print file. For product documentation of a "lite" version of a product that has fewer features than the main product, the higher level features can be included only in the main product documentation.

Using Includes

Includes can be a valuable tool for reusing the same content in multiple places. Content included in many projects only needs to be created once and can be updated in one place.

Boilerplate

Many documentation projects have common files. For example, every document will typical include a copyright page and may include a logo image. DynaDoc allows these files to be stored in a common folder. The script for this document (shown in the previous section) includes the following line:

- !INCLUDE,required/intro_.txt,1

Click on the filename above to view the intro_.txt file.

DynaDoc also automatically includes a script preface, which can contain additional global commands and pages. It is called, "script_preface.txt" and can be located in a convenient location. DynaDoc uses the variable $common to define this location.

Using NUL

Using NUL for the output file in the script will prevent DynaDoc from generating output. This is useful for including files that already exist.

Using NUL for the input file in the script will prevent DynaDoc from creating a link in the menu. A static entry is included instead.

Renaming Output Pages

In general, output files are named after the input file, using .html instead of .htm. In some cases, it is preferable to force the output file to use a different name.

Case 1: If you want to use a page from another project, you can rename the output to exclude the path from the input file and the output will be included in the current document's output structure.

Case 2: If you want to reuse a page more than one time in a document, you can rename the output so that the previous and next links will not be broken. Otherwise the next link on the first occurrence of the page would jump to the page after the last occurrence of the page.

Training Documents

DynaDoc uses two special conditional page identifiers, "i" and "s." The "i" can be used to create companion instructor notes pages for each page and the "s" can be used to create student notes pages.

Working with Multiple Documentation Projects

If the project and project directory are not included in the URL, DynaDoc is designed to read a project list. It then displays a page with all of the projects. Document lists can be categorized allowing projects to be grouped.

The project list is a text file with the following structure:

- <projectDescriptiveName>|<projectName>|<projectDir>|<OutputDir>|<docList>|

The project list should be named, "dynadoc_???.lst" where ??? is a three character reference of your choice.

Additional Notes

DynaDoc can be coupled with other components or applications to create even more advanced documentation.

My_Notes

My_Notes is a JavaScript tool that allows users to add their own notes on a page-by-page basis. The capabilities and end use of My_Notes are described in the Customize this Document section.

Automatic Content Generation

DynaDoc can be coupled with systems that generate automatic documentation. Examples include:

- A Parts Navigator that creates linked pages of parts BOMs using an export of parts data from an ERP system.
- A set of sequential removal and replacement instructions for repair purposes. The unique feature of this system is that the instructions for each part are written only once, even though the file may be used in multiple places because the part must be removed to get to many other parts in the assembly.
- A troubleshooting decision tree exported from a spreadsheet.