



VideoRay Document Creation





Copyright Notice

This material is copyright protected. No material may be reproduced or transmitted in any form or by any means for any purpose without expressed written consent of VideoRay LLC.

Copyright © 2015, VideoRay LLC - The Global Leader in Micro-ROV Technology

Table of Contents

- Copyright
- Table of Contents
- About this Document
- How to Get Help

Process Overview

Environment Configuration

- Text Editor
- Git
- XAMPP
- Additional Utilities

Project Setup

- Git Repository
- Dynadoc Preparation

Content Creation

- DynaDoc Script
- HTM
- Images
- JavaScript
- CSS
- Advanced Features
 - View and Print
 - Document Structure
 - Glossary

Publishing

- Self Extracting Zip
- Posting via FTP

Archiving

- Commit and Push to GitHub





Using DynaDoc Online

About this Document

This document provides information about developing documentation projects from an administrative and implementation perspective. Technical aspects of using DynaDoc can be found in the [DyanDoc Operator's Manual](#).

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 Document Creation 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 (<i>select option 1</i>) |
| Skype | videoray.support (<i>by appointment</i>) |
| Remote Sessions | www.videoray.com/support/remote-support.html (<i>by appointment</i>) |

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 your local VideoRay dealer. We can provide guidance or help you find a certified consultant.

Document Creation Process Overview

The VideoRay document creation process relies on DynaDoc and several supporting PHP programs. In addition, several other free or open source tools are required. The VideoRay proprietary programs, including DynaDoc, are archived in private Git repositories on [BitBucket](#). Other tools can be found on the Internet. These programs can be installed on a local computer to enable a stand-alone document production environment. Information about downloading and installing the required programs and setting up the development environment can be found in the [Environment Configuration](#) section of this guide.

Documents are written in HTML with each page being a modular HTML file. JavaScript can be added to make the pages dynamic. A template and CSS files control the page layout, so the author only needs to concentrate on the content of each page. The sequence of the pages in the final document is defined in the DynaDoc script. The results are in the format of this manual, with an automatically generated menu, table of contents and other features. For more information about the technical aspects of using DynaDoc, see the [DynaDoc Operator's Manual](#).

Documentation projects are version controlled using Git and archived in public repositories on [GitHub](#). Completed documents are published to VideoRay product computers and the Internet.

This guide provides step-by-step instructions on how to create and publish VideoRay documents. The basic process is outlined below and details are included in the sections that follow.

1. Set up the working environment (one time).
2. Set up a document project.
3. Create the content.
4. Prepare the document for publishing.
5. Publish the document.
6. Archive the project.

Environment Configuration

Prior to working on documentation projects, several software utilities need to be installed on your local computer and the environment must be set up.

The default document development folder is: C:\VideoRay\www\. Create the VideoRay folder if it does not exist, but do not create the www folder yet.

The environment requirements include:

- A text or HTML editor
- Web server with PHP
- Git client
- Additional utilities

The following sections provide information about configuring your system for these requirements.

You will also need [BitBucket](#) and [GitHub](#) accounts and to be assigned to the VideoRay Documentation team on each website.

Text Editor

Documentation is written primarily in HTML. Most HTML editors are either expensive or produce poorly formatted results. Several good choices for advanced text editors are listed below:

- [Notepad++](#) (Recommended)
- [EditPad](#)
- [SublimeText](#)
- [Atom](#)
- [Vim](#)

If you have a preferred text editor, feel free to use it instead.

Configure windows to display file extensions.

1. If using Windows 7, open Windows Explorer and click **Organize** in the top left.
2. Select **Folder** and **Search Options**.
3. Navigate to the **View** tab.
4. Uncheck the checkbox next to **Hide Extensions For Known File Types**. This allows you to see and alter file extensions.

Configure your editor to automatically open .txt, .htm and .php files.

1. Create a blank .txt file on your desktop. Rename it and replace the .txt with .htm
 2. Right click the new .htm file and select **Properties**.
 3. In the **General** tab, click **Change** next to the **Opens With** field.
 4. Navigate to the executable for your editor and select it. This will configure all .htm files to open with your editor.
 5. Repeat this process for .php files.
-

Git

Git is a version control and archiving platform. Files and changes to the files are stored in repositories. VideoRay uses private repositories on [BitBucket](#) to store DynaDoc and other proprietary utilities and public repositories on [GitHub](#) to store the document source (The source is the same as the published content, except in a different format, so it does not need to be protected).

Git Client

1. Download and install [SourceTree](#), or any other Git client.
2. Other Git client choices can be found on the [Git](#) website.
3. If you are familiar with using Git via the command line, you can use that as well.

BitBucket

You will need to clone the www repository from BitBucket before you can start creating documents.

1. Visit [BitBucket](#).
2. Create an account if you don't already have one. Your username should be easily recognizable by other employees.
3. Inform your supervisor of your account name, and have your permissions configured for the VideoRay Documentation team.
4. Using BitBucket or SourceTree, Clone the www repository to your local computer in the following location:
C:\VideoRay\www.

If using SourceTree, click on the **Clone/New** button.

1. Enter the **Source Path / URL**:, <https://bitbucket.org/videoraydoc/www>
2. Enter the **Destination Path** using the location from above.
3. You may be prompted for your BitBucket account credentials.

If using BitBucket, navigate to the www repository on BitBucket.

1. Click **Clone** in the **Actions** section of the sidebar (indicated by "...").
2. Click on the **Clone in SourceTree** button.
3. Allow the SourceTree application to launch.
4. Enter **Destination Path** using the location from above.

GitHub

GitHub is required to clone a document set to your local computer for editing and to push documents back to GitHub for version control and archiving.

1. Visit [GitHub](#).
2. Create an account if you don't already have one. Your username should be easily recognizable by other employees.
3. Inform your supervisor of your account name, and have your permissions configured for the VideoRay Documentation group.

At this point, you will not need to access GitHub any further for preparing your environment. You will need to access GitHub when you start to work on documentation projects as explained in the [Document Setup](#) section.

XAMPP

XAMPP provides a suite of products for hosting a local web server on your computer to run DynaDoc and other PHP scripts.

1. Download the latest version of [XAMPP](#) for Windows.
2. Install XAMPP. Do not install XAMPP in a Program Files directory. Instead use C:\xampp.
3. Start XAMPP Control Panel. On the right side, click the **Config** button with the wrench in it.
4. In **Editor**, navigate to your text editor's executable and select it.
5. In **Autostart**, check the box next to **Apache**.
6. Click **Save**.
7. In the main control panel window, click the **Config** option across from **Apache**. Click the first item in the menu that appears, httpd.conf
8. A text file should open in your text editor. There are two lines that need to be changed:
 1. Change
DocumentRoot "C:/xampp/htdocs"
to
DocumentRoot "C:/VideoRay/www"
 2. Change
<Directory "C:/xampp/htdocs">
to
<Directory "C:/VideoRay/www">
Do not remove the < > brackets.
9. Restart Apache for the settings to take effect.

The www repository (which should have been downloaded earlier) has an index.htm file that is set up with links for DynaDoc and other required PHP programs. To access the web server, open a browser and enter <http://localhost> or <http://127.0.0.1> in the address bar. You should see a page with Web tools, Apps, DynaDoc and Glossary links.



You can create an index.php file and include your own links, then at the bottom use the PHP include command to include the index.htm file with the DynaDoc links.

```
<?php
```



```
include "index.htm";  
?>
```

Additional Recommended Utilities

There are several other utilities that you may find useful in developing documentation projects. These utilities are all free.

- [7-Zip](#) - allows for creation of compressed archives and extraction of files from these archives.
- [grepWin](#) - is a search utility that allows searching of phrases in multiple text files. grepWin can search HTML or script files for key strings, and even supports mass find and replace in multiple files. It is very useful if you ever need to change things like directory paths in your files.
- [LinkChecker](#) - can be used to validate all links in your documents and should be used before publishing a document to ensure that all links in the document are valid.
- [GIMP](#) - is an image editing utility for working with document images.

As with all software recommendations, if you have another tool that you are more comfortable with that performs similar functions, feel free to use that instead.

Project Setup

After your environment has been set up, you can start working on documentation projects. The instructions in this guide are for a representative documentation project. The nature of your project may vary and may require implementation of different features or elements.

The steps for working on projects depend upon whether you are creating a new project or editing an existing project.

Creating a New Document Project

1. Create a project repository on GitHub.
2. Clone the project repository to your local system.
3. Configure DynaDoc to recognize the new project.
4. The project is ready for editing.

Editing an Existing Document Project

1. Clone the project repository to your local system.
2. The project is ready for editing.

Git Repository Creation / Cloning

If creating a new document, you need to create a GitHub repository to store a copy of your documentation project online and manage changes to it, and then clone this repository to your local computer for editing.

If you are editing an existing document that you have not yet configured locally, you need to clone the existing repository to your local computer and then you can edit it.

Creating a Document Repository



If you are planning to edit an existing document, skip this section.

1. Navigate to the [VideoRay-Documentation](#) GitHub site. You will need to log in to view it.
2. Click **New Repository** in the top right corner. If you are not logged in or are not an Owner in the VideoRay Documentation group, this option will not appear and you will either need to log in or contact your supervisor.
3. Give the repository a name. The convention for repository naming is:
 1. All letters are lower case.
 2. Spaces are replaced with underscores.
 3. The name should provide a general idea of the document's contents.
4. Include an informative description of your project in the **Description**.
5. Leave the privacy setting on **Public**.
6. Check the box next to **Initialize this Repository with a README**.
7. Click **Create Repository**.

The repository exists on the GitHub server now, but it needs to be cloned to your local machine as well in order to work on it.

Cloning the Document Repository to Your Local Computer.

These steps apply to newly created project repositories as well as for preparing to edit an existing project that you have not yet configured locally.

1. Start SourceTree or your preferred Git client or use the command line. The following instructions assume you are using SourceTree
2. Click on the **Clone/New** button.
3. Enter the **Source Path / URL**: to the repository on GitHub:
`https://www.github.com/VideoRay-Documentation/repositoryName`
(replace *repositoryName* with the name of the desired repository).
4. Enter the **Destination Path** for the local copy of the repository:
`C:\VideoRay\www\repositoryName`
(replace *repositoryName* with the name of the desired repository). You can set up C:\VideoRay\www\ as the default path in SourceTree.
5. Click the **Clone** button.

6. You may be prompted for your GitHub account credentials.

You should now have the repository (including its contents if editing) on your local machine. There are few more steps required to configure DynaDoc to work with this document. These steps are explained in the next section.

Dynadoc Project Preparation

DynaDoc and DynaDoc utilities are included in the www repository. All DynaDoc PHP programs can be accessed through the links or buttons on the home and categories pages.

DynaDoc will already be configured for all of the projects that exist at the time you clone the www repository to your system and will be updated each time you "pull" the www repository to your system. If you want to edit an existing project, you will only need to clone the project repository to your system (as described in the previous section about [Git Repository Creation / Cloning](#)). This only needs to be done once. Then as you make changes to the project, you will "push" your changes to the project repository on GitHub.

If you are creating a new project, in addition to creating and cloning the project (as described in the previous section about [Git Repository Creation / Cloning](#)), you will need to configure DynaDoc to recognize the project and then "push" the www repository to GitHub so that others can keep their system up-to-date with all projects as well.

Configuring DynaDoc for a new project is a semi-automated process which uses the `dynadoc_create.php` program for these steps (`dynadoc_create.php` is included in the www repository and is accessed using the `dynadoc_manage.php` program, which runs when you click on a category link on the home page.). If you are creating a new project, these steps will configure DynaDoc and copy the project template files to the project folder. If you are editing an existing project, you do not need to perform these steps.

1. Ensure that XAMPP is running, and type localhost into your browser's address bar, and press enter. This should load your local home page with the dynadoc category links.
2. Click on the category in which you want the project to reside.
3. You should see a list of various projects in that category.
4. Click **Create New Document Project** at the top. This will display a form for you to fill out.
 - o **Project Type**: Select whether this is a new documentation project or you will be editing an existing documentation project.
 - o **Category**: This will default to the current category.
 - o **Project Name**: This will define the document's title and be displayed at the top of each page.
 - o **Project Folder**: This should be the name of the repository you created on GitHub and cloned on your local computer.
 - o **Output Folder**: This should be left blank unless there are special requirements.
 - o **Document Type**: This should be "Operator's Manual," "Maintenance Manual" or some other description of the type of document you are creating / editing.
 - o **Quick Start**: This setting determines whether or not the document will have a Quick Start section, not have a Quick Start section, or be a blank document.
5. Click **Prepare Documentation Project** when all settings are correctly configured.

The `dynadoc_create.php` program will prepare DynaDoc to work with your document. The next section will describe content creation - for more information about using DynaDoc to create and edit documents, see the following sections and the [DynaDoc Operator's Manual](#).

Content Creation

The content creation process includes edit and review cycles. The summary of steps is:

1. Write the script.
2. Write the htm pages.
3. Generate the document.
4. Review the document.
5. Repeat as necessary.



The information provided here is just the essentials. For more detailed information, see the [DyanDoc Operator's Manual](#).

Now that you have set everything up, you can start to work on your document. First, let's take a look at the provided files and a general structure of the files we'll be working with. A typical project folder will look something like this.

| Documents library | | | | |
|-------------------|----------------------|---------------|------|--|
| pam | | | | |
| Name | Date modified | Type | Size | |
| .git | 7/29/2014 10:49 A... | File folder | | |
| glossary | 7/25/2014 12:06 PM | File folder | | |
| htm | 7/31/2014 1:33 PM | File folder | | |
| images | 7/23/2014 2:40 PM | File folder | | |
| javascript | 7/23/2014 10:38 A... | File folder | | |
| My_Notes | 6/5/2014 8:36 AM | File folder | | |
| .gitignore | 7/25/2014 12:06 PM | Text Document | 3 KB | |
| pam.txt | 7/23/2014 2:39 PM | TXT File | 1 KB | |
| README.md | 6/5/2014 8:35 AM | MD File | 1 KB | |

- **projectName.txt**: This file controls how dynadoc reads and generates the document. We will discuss it in depth in the following sections.
- **css**: This folder can contain any local css required.
- **glossary**: The glossary folder contains two files, a `glossary.csv` and an `exclude.csv`. These files are used to automatically generate a glossary for your document, is covered in the DynaDoc manual.
- **htm**: This folder contains the raw modular htm files that are the actual content of the project. HTML files are simply HTML. In DynaDoc, the `.htm` extension is used to designate that the file is input into the DynaDoc scripts, while HTML files are the output. Most of our time will be spent working with files in this folder.
- **images**: This folder contains local images that will be included in the document.
- **javascript**: This folder contains local javascript files. Javascript is a programming language that allows us to make HTML pages more dynamic. Discussing JavaScript's many applications is beyond the scope of this document, but several basic examples will be provided.
- **My_Notes**: This folder contains files used by an end-user to create custom notes in the document. We will not be making any considerable changes to the files in here.



The required folder (installed as part of the www repository) is to be used for any common elements, such as images for the logo and page navigation, or local css. The local folders should only be used for project specific content.



The template folder (installed as part of the www repository) is used to create new projects. Git does not include empty folders, so the template will not include all folders above. You can add desired folders like images and pdf to each project individually, or you can add folders to your local template folder so that all new projects you create will include these folders by default.

DynaDoc Script

The DynaDoc script is a text file that you create in the root directory of your project. It defines some document parameters and the sequence of pages of the document. The DynaDoc script is described in more detail in the [DynaDoc Operator's Manual](#).

The DynaDoc script for this document is shown below.



The display below is dynamically included as part of the document generation process using the `<!-->` and `<!-->` tags and will reflect the state of the script file when this document was generated.

```
#Document Commands
!PRODUCT,Document Creation
!VERSION, 1.00.00
!DOC_TYPE,Guide
!OUT_DIR,document_creation/
!VALID_PAGES,
!VALID_TEXT,
!TEMPLATE_COVER,required/htm/template_index_print.htm,index_print.htm

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

*,Process Overview,,overview_.htm,,,
*,Environment Configuration,,environment_.htm,,,
**,Text Editor,,environment_text_editor.htm,,,
**,Git,,environment_git.htm,,,
**,XAMPP,,environment_xampp.htm,,,
**,Additional Utilities,,environment_additional.htm,,,

*,Project Setup,,project_setup.htm,,,
**,Git Repository,,project_setup_git.htm,,,
**,DynaDoc Preparation,,project_setup_dynadoc.htm,,,

*,Content Creation,,content.htm,,,
**,DynaDoc Script,,content_dynadoc.htm,,,
**,HTML,,content_html.htm,,,
**,Images,,content_images.htm,,,
**,JavaScript,,content_javascript.htm,,,
**,CSS,,content_css.htm,,,
**,Advanced Features,,advanced_features.htm,,,
***,View and Print,,view_print.htm,,,
***,Document Structure,,document_structure.htm,,,
***,Glossary,,content_glossary.htm,,,

*,Publishing,,publishing.htm,,,
**,Self Extracting Zip,,publishing_zip.htm,,,
**,Posting via FTP,,publishing_posting.htm,,,

*,Archiving,,archiving.htm,,,
**,Commit and Push to GitHub,,archiving_git.htm,,,

*,Using DynaDoc Online,,dynadoc_online.htm,,,
```

This may look a bit complex, but there's really not much we need to worry about. Let's start with the first section, under `#Document Commands`. Most of these were already set when we ran the script to create the configure DynaDoc for the project. The only options we need to worry about for a basic document are the `PRODUCT`, `VERSION` and `DOC_TYPE`.

- **PRODUCT:** This value plus `DOC_TYPE` control the title of the document, and will be separated with a space. It will be set when you create the project, but can be edited here.
- **VERSION:** The number at the bottom of every page indicates the version number of the document. You should update this as you make changes, with development versions being decimals such as 0.7.00, initial public releases usually numbered as 1.00.00, and versions that have been updated after initial release having greater values, such as 1.01.00
- **DOC_TYPE:** This is the type of document, such as "Operator's Manual." It is used in the title.

Under `#Documentation Generation`, the script starts with two `!INCLUDEs`. These are used to include boilerplate text to the start of each document. The second `!INCLUDE` will typically be `!INCLUDE,required/overview_.txt,1` if the document does not have a Quick Start section, and `!INCLUDE,required/overview_qs_.txt,1` if it does.

Now that the basic settings are out of the way, the next section determines the content of the document. Each page of the

document requires a separate line with the following fields.

***.title**,outName,**htmName**,jsName,sbName,imgName

Each field page is separated by a comma, and for most pages only three fields are required (these are shown in bold underline above). These three fields are described below.

- *: The number of asterisks at the beginning of a page declaration determine its level in the document hierarchy. A single asterisk is a primary topic, with a two asterisk page being a sub topic under it, and a three asterisk page under that, and so on. DynaDoc currently supports seven topic levels, but the code could be modified to support more if necessary. The hierarchy of this document is fairly flat and only has primary topics and two levels of subtopics, so let's take a look at a sample with a little bit more depth. The example below has one primary topic, under which are two sub-topics. The first sub-topic also has three sub-topics of its own, of which one of those has another sub-topic and is four levels deep. The second-sub topic has no sub-topics. In general, if your pages start becoming very long, you should consider splitting them into multiple pages using sub-topics.

```
* ,Equipment Guide,,equip_.htm,,,
** ,ROV,,equip_rov.htm,,,
*** ,ROV Connections,,rov_connections.htm,,,
**** ,ROV Pin Out,,rov_pin_out.htm,,,
*** ,Buoyancy,,rov_buoyancy.htm,,,
*** ,Propulsion,,rov_thrusters.htm,,,
** ,Optional Utilities,,environment_optional.htm,,,
```

- **title**: The title is used in the table of contents, which is automatically generated, and for the heading on the page.
- **htmName**: This points DynaDoc to the content file source in your htm folder to build the document.



The additional fields are optional. Make sure that your page definition is formatted with the commas as shown, or DynaDoc will not be able to read your script file correctly.

HTM Files

HTM files represent the body of the document. DynaDoc takes the information within the HTM files as directed by the script file, and puts them into a template to make everything look nice for the website.



.htm is used for input source files, and DynaDoc will convert the output file name automatically to .html.

HTML is a markup language, and usually uses an opening sequence and closing sequence of unique characters to tell a browser how to display text. We'll only cover a few basic HTML tags here. If you'd like to know more HTML tags to improve the appearance of your documents, try [W3Schools HTML Tutorial](#).

- **Headers**: A Header makes text larger and automatically gives a bit of space under it. Headers shouldn't be used to enlarge text for general use, instead use them for page titles and sub-sections. A smaller header number is larger.
 - Example: <h1>Writing HTML</h1> or <h3>Writing HTML</h3> for a smaller version.
- **Paragraphs**: Paragraph tags format text properly so that it wraps correctly on all displays. Generally if a text has no other formatting tags on it, it should be wrapped in paragraph tags to ensure proper formatting with Dynadoc.
 - Example: <p>I'm a paragraph! Use me for good formatting!</p>
- **Bold and Italics**: Old favorites.
 - I'm Bold! and <i>I'm Italic!</i>
- **Web Links**: Clicking on the text in one of these will take the viewer to the webpage you define.
 - Example: VideoRay
 - Here's an example of an internal link. This link would take a user to this page.
VideoRay
- **Images**: Images will be discussed in more detail in the [Images Section](#).
- **Lists, Ordered and Unordered**: A slightly more complicated list. Ordered lists will number every entry in them automatically, while unordered lists will display bullet points. We're actually in an unordered list right now!
 - Example: An unordered list

```
<ul>
  <li>I'm the first bullet point!</li>
  <li>I'm another bullet point!</li>
  <li>I'm the last bullet point!</li>
</ul>
```

- Example: An ordered list

```
<ol>
  <li>I'll be numbered 1!</li>
```

```
<li>I'll be numbered 2!</li>
<li>I'll be numbered 3!</li>
</ol>
```

There are many more HTML tags, but these should get you started.

Recommended HTML Conventions

All tags should be in lower case.

All `<h>`, `<p>`, ``, etc. tags should begin on a new line.

`` and `` tags should also begin on a new line and be indented two spaces for each level.

Heading order is `<h1>`, `<h2>`, `<h3>`, ``

When referencing local images, the following format should be used. `../projectName/images/imageName`.

When referencing local links, the following format should be used. `../projectName/html/htmlName`.

Images

Including an image in an HTML page is fairly simple, but first we need one to include. The most likely images you'll need while making a document are pictures of relevant hardware, or screenshots of various software elements. We may need to edit the images a bit as well, but let's start with acquiring them.

Taking a Picture

While taking a photograph for a document, ensure that you have good lighting and a uniform, preferably white, background. This makes removing the background and any further editing we need to do to the picture easy. Take several photographs of each piece to ensure that there is at least one crisp picture to use. Good backgrounds and clear pictures with good lighting are critical to maintaining the professional appearance of a document.

Uploading a picture to your computer will vary depending on your camera, but plugging the camera into your computer will normally allow you to open its internal storage. Source images should be stored in the projects source folder.

Taking a Screenshot

We can also use the keyboard's "Print Screen" button to capture an image of a computer screen. This is especially useful if you need to take a screenshot of a software program. Bring up the item you'd like to capture, and then press the Print Screen button (which may vary by keyboard). This will capture the screen, which you can then paste into an image editing program like GIMP or Paint. Source images should be stored in the projects source folder.

Editing

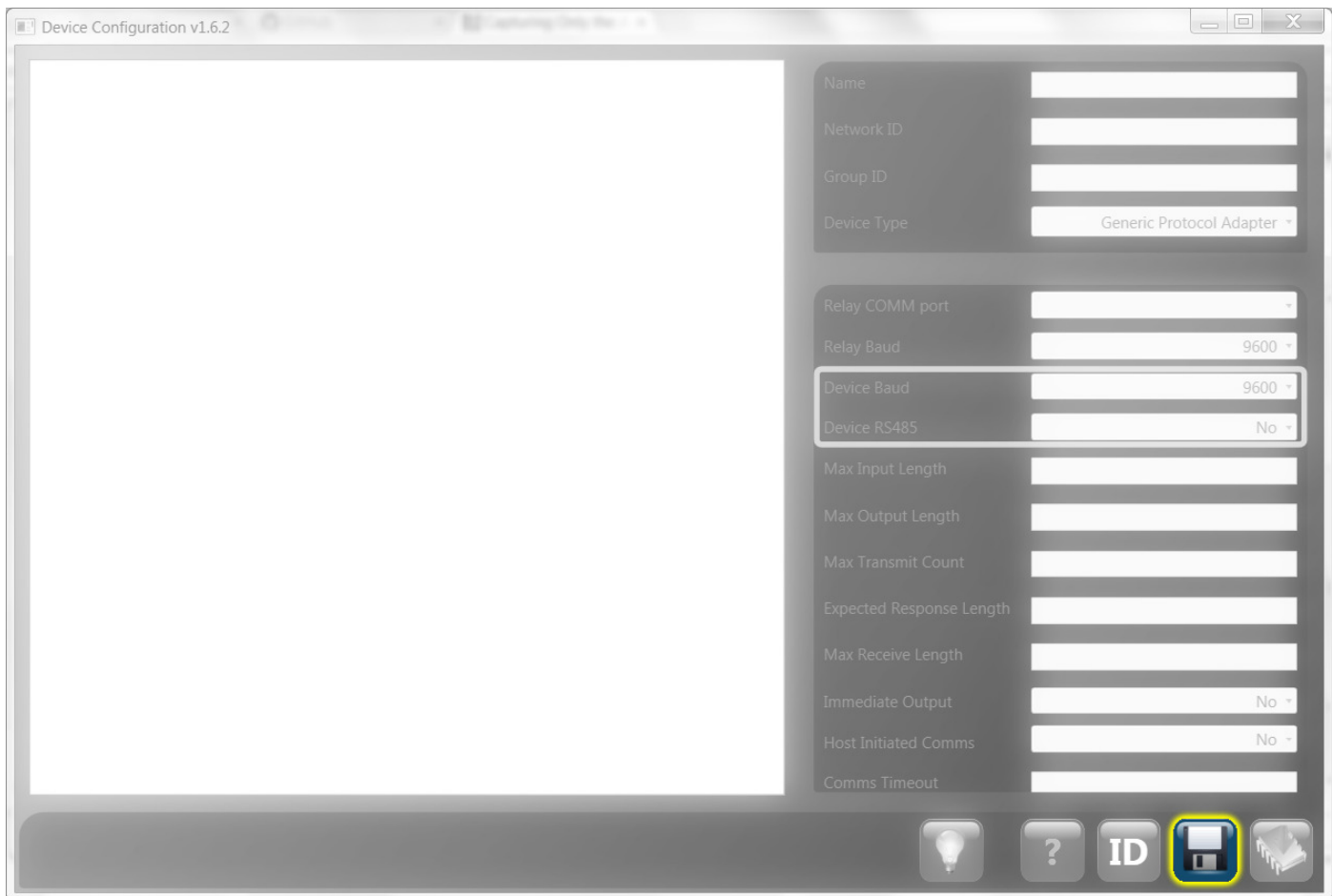
There are two primary edits you'll want to do perform on images, cropping and highlighting.

Cropping

In GIMP, the crop tool allows you to select an area and cut away everything else. It appears as a small razor in the toolbar.

To use it, select an area. After you make your initial selection you can pull on individual corners of your selection box to get a cleaner crop. When satisfied with your selection, press enter to crop the image.

Highlighting



Highlighting allows us to use the shadow and color functions of GIMP to create a nice looking highlight around a section of an image we'd like to emphasize. Image highlighting in all documents should remain consistent, so there is a specific process for this.

1. Open the image in GIMP
2. Using the rectangular selection tool, select the section you'd like to highlight as closely as possible.
3. Press "Shift+Q" on the keyboard. This will activate Quick Mask mode, which shows the non-selected parts of the image covered in red.
4. Use the pencil tool with a size of 1 to paint in individual pixels to get a more precise selection.
5. Press "Shift+Q" again when finished with your selection to toggle Quick Mask off.
6. Press "Ctrl+I" to invert your selection to select all of the image except your object.
7. Go to "Filters" in the top bar, then to "Artistic", then click on "Softglow."
8. Set glow Radius to 50.00, Brightness to 1.00, and Sharpness to 0.00. Click Ok.
9. Go to "Colors" in the top bar, then select "Desaturate." Select Choose shade of gray based on: Lightness, then click ok. This will desaturate the colors of everything besides your object.
10. Press "Ctrl+I" again to invert selection back to your object.
11. Go to "Filters", "Light and Shadow", and click "Drop Shadow."
12. Set Offset X to 0, Offset Y to 0, Blur Radius to 10, and the color HTML notation to "f0f0f0". Click Ok.
13. Press "Ctrl+F" five times to duplicate the shadow to achieve the desired brightness.

Saving

Saving in GIMP is a bit different than usual for image files. By default the image will save a .xcf file, which contains all of the metadata about how the image was edited. This is useful if you need to edit the image again at some time. If you are ready to use the image, instead you will want to go to "File" and then to "Export As." Export your image as a .png to the project's images folder. Keep the default settings for the GIMP prompts and click **Export**.

Including an Image

Now we have an image, but it's not in the webpage yet. The format to include an image is another HTML tag, similar to the ones described in the last section. The tag below was used to display the image on this page.

```

```

The src field specifies the image location. In this case, replace "document_creation" with your own project folder, and replace "con_save.png" with your own image name. Title specifies mouseover text. Width is the width of the image in pixels. There is also a height attribute, which controls the height of the image in pixels. If only one of these attributes is set, the image will

retain its aspect ratio as it is resized.

Javascript

JavaScript is a useful programming language that we can leverage to create more interactive and dynamic web pages. While the instructions presented here are fairly simple, they are recommended for document writers that are a bit more tech savvy, rather than general use. In the example here we'll be creating text that changes an image when the text is moused over, and then changed back when the mouse leaves the text. Our implementation will be split into a JavaScript and HTML file, and will then be put together by DynaDoc.

JavaScript File

If you don't already have one, create a folder named javascript in your project directory. In the javascript folder, create a text file. Rename the text file to have the same name as the HTML page you want to include JavaScript on. The file extension for JavaScript files is .js. If your HTML page is "software_cockpit.htm", your javascript file should be "software_cockpit.js".

We'll start with a useful example. We'll use JavaScript to preload some images and then have an image swap between different highlights when the relevant text is moused over. Let's take a look at the JavaScript file first.

```
//VideoRay
if (document.images)
{
    base = new Image
    view = new Image;
    enumerate = new Image;
    gid = new Image;
    rs = new Image;
    consav = new Image;
    condl = new Image;

    base.src = '../images/con_view.png';
    view.src = '../images/con_view.png';
    enumerate.src = '../images/con_ques.png';
    gid.src = '../images/con_id.png';
    rs.src = '../images/con_rs.png';
    consav.src = '../images/con_save.png';
    condl.src = '../images/con_dl.png';

    .....

    function reset()
    {
        .....
        document.view.src = base.src;
    }
}
```

Not a lot of text, but most of it doesn't make sense on its own. Let's explain what's going on.

- **//VideoRay**: In JavaScript, comments are indicated with two forward slashes. We won't read this line as code, but it's useful for searching files to have it in there.
- **if (document.images)**: This is a short check to see if there are any images in the document at all. If there are, it executes all of the indented code.
- **[name] = new Image**: These lines create images and assign them names. The images don't actually have any information in them besides names, but we'll get to that later. You'll need at least three images. Base will be the image that the page defaults to when no text is moused over. View will be the current image being displayed, and will therefore have the same value as base to start. You'll also need at least one image that you'll actually be swapping to.
- **[name].src = '[imagelocation]'**: This assigns the actual image data to the names we created earlier. View and base should be the same as mentioned above.
- **function reset()**: This is a simple function that sets the currently disabled image back to the base one.

Make sure that you set the names and paths correctly. Next, we'll take a look at the HTML file that implements the JavaScript.

HTML File

```
<h2>Device Configuration</h2>
<p><img src = "../required/images/icon_note.gif"*/> Note that in most cases the factory will configure the PAM for customer use, and users will not need to follow these steps to configure their devices. If you do need to configure your device yourself, please continue.</p>
<p>The baud rate and the choice of RS485 or RS232 can be configured using the vrDeviceConfiguration tool. These settings are stored in EEPROM on the PAM. Before beginning, make sure no other devices are connected to the PAM, and only attempt to configure one PAM at a time.</p>
<p align = "center"><img src = "../pam/images/con_view.png" width = "550" height = "400" name = "view"></p>
</>
<!-- Connect the PAM to the RV and power on everything.-->
<!-- In the VideoRay Cockpit installation directory there will be a file named vrDeviceConfiguration.exe. Double click it. (The default cockpit installation directory is C:\Program Files (x86)\VideoRay\vrCockpit-->
<a name = "enumerate" onmouseover = "view.src=enumerate.src" onmouseout = "reset()"> The Device Configuration utility will open. Once you have verified the PAM is receiving power, click the "P" button to enumerate the PAM.<img src = "../required/images/icon_hand.gif"*/></a>
<a name = "id" onmouseover = "view.src=id.src" onmouseout = "reset()"> The default setting for the PAM is a generic protocol adapter which should not be changed. Click the "ID" button to identify the PAM as such.<img src = "../required/images/icon_hand.gif"*/></a>
<a name = "rs485" onmouseover = "view.src=rs.src" onmouseout = "reset()"> To set the PAM to be RS485 capable, click the dropdown menu next to "Device RS485" and select "Yes". If you are using RS232, leave this setting at "No".<img src = "../required/images/icon_hand.gif"*/></a>
<a name = "baudrate" onmouseover = "view.src=baud.src" onmouseout = "reset()"> To set the Device Baud, click the dropdown menu next to "Device Baud" and click the setting your device uses.
<img src = "../required/images/icon_hand.gif"*/></a>
<a name = "consav" onmouseover = "view.src=consav.src" onmouseout = "reset()"> Press the Save Button to save the Accessory Device file for cockpit to use.<img src = "../required/images/icon_hand.gif"*/></a>
<a name = "condl" onmouseover = "view.src=condl.src" onmouseout = "reset()"> Click the download button.<img src = "../required/images/icon_hand.gif"*/></a>
</>
```

This might look a bit more intimidating at first glance, but most of it is just regular text. Let's take a look at the relevant parts for the image.

- **<p align = "center"></p>**: Here the image that will be replaced and it's base state are defined. The image has been named view to match the syntax of the JavaScript file, and the width and height have been manually set. Manually setting the size ensures that the image remains the same size even when it's swapped to a new image. This avoids shifting text around on the page as the image changes in size. The image is also wrapped in paragraph tags with align="center" to center the image on the page.
- **[Link Text]**: Here's an example of the text that actually executes what we want. This creates a hyperlink that, instead of taking you somewhere when you click it, does things on mouseover and on mouseout. In this case, we've set it so that the view image has it's source changed when the mouse is over the text, and have it call the reset function we defined earlier when the mouse leaves. Just make sure to replace the relevant fields with the appropriate names, and an interactive page is just about ready.

DynaDoc Script File

We separate the HTM and JavaScript files for ease of use, but we need to combine them to display a proper webpage. DynaDoc is built to do this for us. One of the fields we discussed earlier is used to include a JavaScript file. Here's an example of a page with an included JavaScript file in the DynaDoc script.

```
**Device Configuration,,software_device.htm,software_device.js,,
```

The JavaScript include is one field after the htm file. If the file has incorrect comma placement it will not read correctly, so make sure to double check the syntax.

CSS

CSS defines page style characteristics. There is a default CSS set in the required folder. If you want to use project specific CSS, you can add it to the project's css folder.

Advanced Features

DynaDoc is a powerful program with a lot of advanced features. The following section describes a few of these features. See the [DynaDoc Operator's Manual](#) for more information about these and additional features.

View and Print

DynaDoc can create view and print versions of a document simultaneously from a single source. This allows you to improve the user's reading experience depending upon whether they are viewing the document online or in print.

Document Structure

DynaDoc is very flexible when it comes to a document's structure. You can use "includes" to replicate boilerplate text or pull document pages from other documents to include in your current document. You can also use conditional text and

conditional pages to create different outputs from the same source.

DynaDoc also has provisions for including sidebar content (below the menu) and images for the menu and table of contents.

See the [DynaDoc Operator's Manual](#) for more information.

Glossary

DynaDoc includes a script that will automatically generate a glossary of terms page with links to pages where the terms appear. You need to define the terms and definitions.



Generally, the glossary is updated only when the script is ready for publishing. You do not need to update the glossary links during the normal edit and review cycles.

Building the glossary is fairly easy, and uses very basic syntax. The glossary source should be created as a text file in the glossary folder and named `glossary.csv`. Each line contains the following elements separated by "|". The trailing "|" is required.

```
case_control|word|description|
```

Case Control can have the following

- * - Forces case sensitive match, and can be used for acronyms.
- ~ - Separates variations of words, example: "video ~video, ~video." will not match VideoRay

If case control is not used, the "|" is still required before the word.

To create a glossary, follow these steps:


1. open `glossary\glossary.csv` in your editor.
2. The first line in the file should be: **case_control|word|description|**
3. After this required line, we can start adding entries. Entry format is always `|Keyword|Definition or Description|` as shown in the above image.
4. Create a new line after every glossary entry.
5. Once you've saved your file with the entries, we need to run the `dynadoc_glossary.php` script using the **Glossary** button.
6. Open `localhost`, and navigate to your project's category.
7. Click on the "Glossary" button to create the glossary.
8. The glossary is generated.
9. Make sure to regenerate your document with the regular DynaDoc script to include the newly created `_glossary.htm` file.



`exclude.csv` provides the glossary script with a list of pages that should be ignored when the glossary links are created. For example, links should not be created to the table of contents. A master `exclude.csv` is included in the required folder. If you want to exclude pages on a project basis, create an `exclude.csv` file in the project's local glossary folder. See the required `exclude.csv` for the file's format.

Publishing

After completing the edit and review cycles, we have to get our document into a state where others can actually access it. The following steps are required.

1. Set the version number for this release.
2. Run LinkChecker to validate links (fix any broken links and rerun dynadoc.php).
3. Generate the glossary using dynadoc_glossary.php using the **Glossary** button.
4. Delete the contents of the html folder to clean up any unused output files.
 **Make sure you do not delete the contents of the htm folder!**
5. Run dynadoc_orphan.php using the **Find Orphans** button to check for unused (orphan) .htm files and clean up and unnecessary files.
6. Generate the document using dynadoc.php using the *projectName* button.
7. Create the pdf of _print.html.
8. Use dynadoc_publish.php using the **Publish** button to create the distribution set of files. On the main DynaDoc page where the button to generate the document is, there is a "Publish" link. This will copy html, images, javascript, and other output folders and files to a publish directory. By default this will be in www/publish.
9. Create the self extracting zip file.
10. Post the distribution set to the desired locations

Self Extracting Zip

A self extracting zip file makes the document set more portable.

1. Navigate to the DynaDoc publish folder.
2. Right click the document folder and select **7-Zip->Add to Archive**.
3. Set the **Archive** name to the product folder name.
4. Select **7z** as the **Archive Format**.
5. Check the **Create SFX archive**.
6. Click on the **Okay** button to create the exe zip file.
7. Run dynadoc_zip.php using the **Zip** button.

Posting

The publish file set should be posted to the following locations:

- The public site: <http://download.videoray.com/documentation>.
- The rovdoc site: <http://www.rovdoc.com>.
- The build set for production systems.

Posting can be completed using any FPT client.

Archiving

The project repository should be updated to archive the current version of the project.



Every so often when working on a project, you should "push" your changes to GitHub so that others can be up-to-date on your work. You can ""push" your files to the project repository as often as you feel necessary to preserve the working state of the project and in case your system suffers from a hard drive or other failure.

Commit and Push to GitHub

All projects should be "pushed" to the GitHub repository when completed.

1. Open SourceTree (or use your preferred git method).
2. On the left sidebar, double click the name of your document project.
3. In the project window, check the status of the files and update `.gitignore` if necessary to remove any unwanted files from being included in the repository.
4. Stage all of the changes.
5. Enter the version number and any additional desired notes in the Commit Message section.
6. Click on the **Commit** button.
7. Click the **Push** button at the top.

Dynadoc Online

A newer feature of dynadoc is the ability to edit the content of individual pages online. This feature is ideal for users who only need to edit page content rather than create their own documents. For example, to translate a document now, all you need is a web browser. The full version can be accessed at <http://www.rovdoc.com/developer/>.

The following descriptions apply to the <http://www.rovdoc.com/developer/> page.

Preprocess the Maintenance Manual

This option loads the display order for maintenance manual documents. To use this function, enter your working folder(ex: korean, german) and click preprocess. The script will take a few seconds to run, and will then display how the order was generated.

Generate a Document

This options runs DynaDoc on a script, generating the output. Enter the name of your working directory in workspace folder, and enter the name of your dynadoc script, such as pro_4, into the project script field. This will generate your DynaDoc output normally, and also add a link to view your newly generated document at the bottom of the page.

There is also a field for advanced options.

- **Project Folder:** The folder containing the project script, if different from the workspace folder.
- **Output Folder:** The folder for the output, if you wish to change it from the default.
- **Use Project List:** Loads documents from a list.
- **Use Edit Mode:** When this option is checked the script will generate pages with a built in edit button. This allows you to call the editor for the page right from viewing it. Make sure to uncheck this option for your final draft.

Edit a Document Source Set

This option is the primary use of our online scripts, and enables the actual editing of documents. Edit your workspace folder and project script in the fields, and click on the edit button to begin the script. The script will output a series of links. Clicking on one of these links will open an HTML editor for that page. When you are finished, click the save icon at the top of the editor to commit the changes to your working folder. Some sections of a document may not produce links. These are documents that are actually located in separate areas and are only linked by the primary document. They must be edited separately.

View Document

This is the simplest option, and lets you view a generated document. It's recommended that you view any pages that you alter to make sure they display as intended.