

# Minimalism – minimalist documentation

*A research note by Namahn*

---

## Introduction

The purpose of this research note is to introduce minimalism as a new approach to documentation design.

---

## Definition

Minimalism is a design philosophy. The theoretical foundation for the minimalist approach to designing instruction and documentation was laid out in 1990 by John M. Carroll in the **Nurnberg Funnel**. Basically, minimalist theory says to get users going right away, minimize reading and make them fill in the gaps themselves. This is why minimalist documentation will often contain procedures with little or no introduction and explanation.

The “paradox of sense-making” is considered the basis for minimalism: when people learn to use computers, they attempt to make sense of the way the system behaves. They make inferences, perform experiments, and look for cause-and-effect relationships. What they do not do is follow step-by-step directions very well. On the contrary, they resist detailed systems of instructional steps and avoid careful planning.

The **main minimalist techniques**, described in the Nurnberg Funnel, are:

- Train users on real tasks (task orientation)
- Provide action-oriented learning (people want to learn by doing)
- Help users get started fast (aim: to sustain interest and activity)
- Guide discovery and exploration
- Rely on the user to think and improvise
- Exploit the user’s prior knowledge and experience
- Slash the verbiage (brevity - “less is more”)
- Make use of incomplete descriptions and instructions:
  - fading technique
  - layering as a safety net for minimalist documentation: layering means providing extra information via pop-up windows, tabs, buttons... cf. inline help
- Support error recognition and recovery
- Allow for reading in any order by providing modular units of information (cf. help philosophy)

These principles challenge traditional, or systems-oriented, approaches to instructional design and delivery. Windows 95’s Help is often quoted as an example of minimalist writing.

---

# Principles and Heuristics for designing minimalist instruction

What follows is an overview of the four major principles and their corresponding heuristics, as presented by Carroll and van der Meij. An elaborately commented version of this overview can be found in the appendix at the end of this note.

- Principle 1: Choose an action-oriented approach
  - Heuristic 1.1: Provide an immediate opportunity to act
  - Heuristic 1.2: Encourage and support exploration and innovation
  - Heuristic 1.3: Respect the integrity of the user's activity
- Principle 2: Anchor the tool in the task domain
  - Heuristic 2.1: Select or design instructional activities that are real tasks
  - Heuristic 2.2: The components of the instruction should reflect the task structure
- Principle 3: Support error recognition and recovery
  - Heuristic 3.1: Prevent mistakes whenever possible
  - Heuristic 3.2: Provide error information when actions are error prone or when correction is difficult
  - Heuristic 3.3: Provide error information that supports detection, diagnosis, and recovery
  - Heuristic 3.4: Provide on-the-spot error information
- Principle 4: Support reading to do, study and locate
  - Heuristic 4.1: Be brief: don't spell out everything
  - Heuristic 4.2: Provide closure for chapters

---

## Opinions on minimalism

### Minimalism as a trend (Teich)

Minimalism fits in with the trends currently shaping the software industry:

- Software is going international as never before, and international documentation has always been minimalist. The instructions are all graphics or, if there is text, it is telegraph-style bites in multiple languages.
- Software is going online in a major way, which means technical writers have to squeeze everything down. At first glance, one could mistake a minimalist manual as online documentation that's been printed.
- A critical mass of computer exposure hit the culture. We no longer have to explain as much as we used to.
- Documentation for many products has reached encyclopaedic proportions. The pressure to cut has been growing.
- Software nowadays is more simple and reliable, thanks to the increase in computing power.

## Theoretical gaps in minimalism (Kearsley)

- Failure to address the social context of learning
- Failure to link itself with other major cognitive and instructional theoretical frameworks
- Failure to address the instruction of other than novice users, i.e. users who are unfamiliar with both the application and the tool
- Lack of knowledge of how well minimalism works for sophisticated computer-using tasks (e.g., aviation, medicine, etc.). Most of the research associated with minimalism has focused on routine tasks – how to use the basic functions of a program.
- Failure to consider individual differences and cultural differences between learners
- Failure to address how minimalism applies to different media, particularly printed documentation versus online systems

---

## References

John M. Carroll (editor): 'Minimalism beyond the Nurnberg Funnel' (articles by Carroll, Van der Meij, Draper, Kearsley)

Mary Ann Eiler. [Minimalism and Documentation Downsizing: The Issues and the Debate](#)'

Phil Teich: '[Is Minimalism More or Less?](#)'

Cecilia G. Ramos: '[Practicing Minimalism: General Overview](#)'

---

## Appendix: Principles and Heuristics for designing minimalist instruction (Carroll and van der Meij)

### Principle 1: Choose an action-oriented approach

#### Heuristic 1.1: Provide an immediate opportunity to act

Documentation often begins with an explanation of how the application and instruction work or an orientation to the semantics of the domain. An alternative (minimalist) approach is to begin by giving the user less to read but more to do. Users are never convinced by talk alone...

#### Heuristic 1.2: Encourage and support exploration and innovation

Users should always feel in control of their own activities. People are more engaged by self-directed activity; they prefer it and learn more from it. Therefore, try to find the optimal balance point between instructing and supporting exploration and innovation:

#### Heuristic 1.3: Respect the integrity of the user's activity

Instruction and documentation must respect the integrity of the user's activity. This will mean subordinating explicit instruction to the continuity of the user's project-oriented activity. So, document designers must often step aside a bit. They can create help but should not impose this on users.

### Principle 2: Anchor the tool in the task domain

To the greatest extent possible, tasks should be selected from the core tasks of the application domain. The users' interest in and understanding of these tasks is what motivated their original interest in the application. Building the instruction from within such tasks capitalizes on this original motivation and helps to satisfy it.

Remember: for most users, an application is a tool they want to use to achieve objectives in the task domain for which that application has been designed. The tool is merely a means. However, instructions are often written as if the tool itself were the user's principal objective.

#### Heuristic 2.1: Select or design instructional activities that are real tasks

Present the user with instruction activities that can instantly be recognized as genuine.

#### Heuristic 2.2: The components of the instruction should reflect the task structure

Concerning the structural organization of a manual, the headings should give an overview of the main tasks in a domain. Such headings help the user keep in view the big picture of the skill being learned. Headings that reflect the task structure may support users in different ways: they can offer scenarios that users otherwise (must) create themselves, support the different points of view users (should) take in task execution, and help users locate information easily when the manual is consulted for reference.

### Principle 3: Support error recognition and recovery

Reducing mistakes and streamlining detection, diagnosis and recovery will make the learning more productive and less frustrating. In general, include error information at a rate of about once for every three actions.

### **Heuristic 3.1: Prevent mistakes whenever possible**

Help users avoid making mistakes by:

- Including hints in the manual
- Using the results from usability tests to rewrite certain sections

### **Heuristic 3.2: Provide error information when actions are error prone or when correction is difficult**

This will occur because of a mismatch with the users' knowledge or because the required actions deviate from a standard routine.

### **Heuristic 3.3: Provide error information that supports detection, diagnosis, and recovery**

After committing a mistake, users need to recognize and locate their error, understand and analyse it to an appropriate extent, and then take some corrective or evasive action.

### **Heuristic 3.4: Provide on-the-spot error information**

Proximal positioning of the error information is crucial: error information should be placed where users need it most: as near as possible to the wrongly executed actions or methods.

## **Principle 4: Support reading to do, study and locate**

Users do not systematically process their manual from beginning to end. Instead, their behaviour is more flexible:

- Sometimes they read to study
- Sometimes they read to locate some information
- Most of the time, they read to do; that is, their reading is task and action oriented

A manual should somehow support all of these users and all their reading strategies.

### **Heuristic 4.1: Be brief: don't spell out everything**

By not explaining everything, a manual stimulates users to activate relevant prior knowledge and depend more on their own thinking. Guidelines:

- If explanations are needed, they should be short to give users the impression of being easy to work through; moreover, try to give them after rather than before task completion
- Create chapters of two to four pages to give users the impression that working them through does not require a large amount of time or the endurance of a long-distance runner (around 20 minutes per chapter works well)
- Use pointers and prompts to have users infer certain facts or procedures. Omit information that can be inferred easily.
- Don't give full screen information

### **Heuristic 4.2: Provide closure for chapters**

Make chapters as independent as possible by providing closure. Closure means starting and ending with a clean desk. To achieve closure, writers should:

- Decide on a home base to start and end each chapter
- Avoid dependency of products across chapters: don't design a complete manual around a single case, the contents of which become more and more complex as users work through chapters

- Avoid dependency of process skills across chapters as much as possible; don't be afraid of repeating certain essential task frequently throughout different chapters. Such repetitions make users fast at and very comfortable with these tasks