Kernel documentation

Jonathan Corbet
LWN.net
corbet@lwn.net

Some 2023 developments

Arch docs move to Documentation/arch
Integration of Rust docs
Spanish translations
Switch to the “Alabaster” theme
Rendering docs into the texinfo format
Rewritten top-level index.rst

Lots of docs written!

What next?

Create Documentation/devices

- Put device-related docs there (~31 subdirectories)

Structure!

Plain text v. HTML

Stats from docs.kernel.org

150,000 requests/day

Top docs:

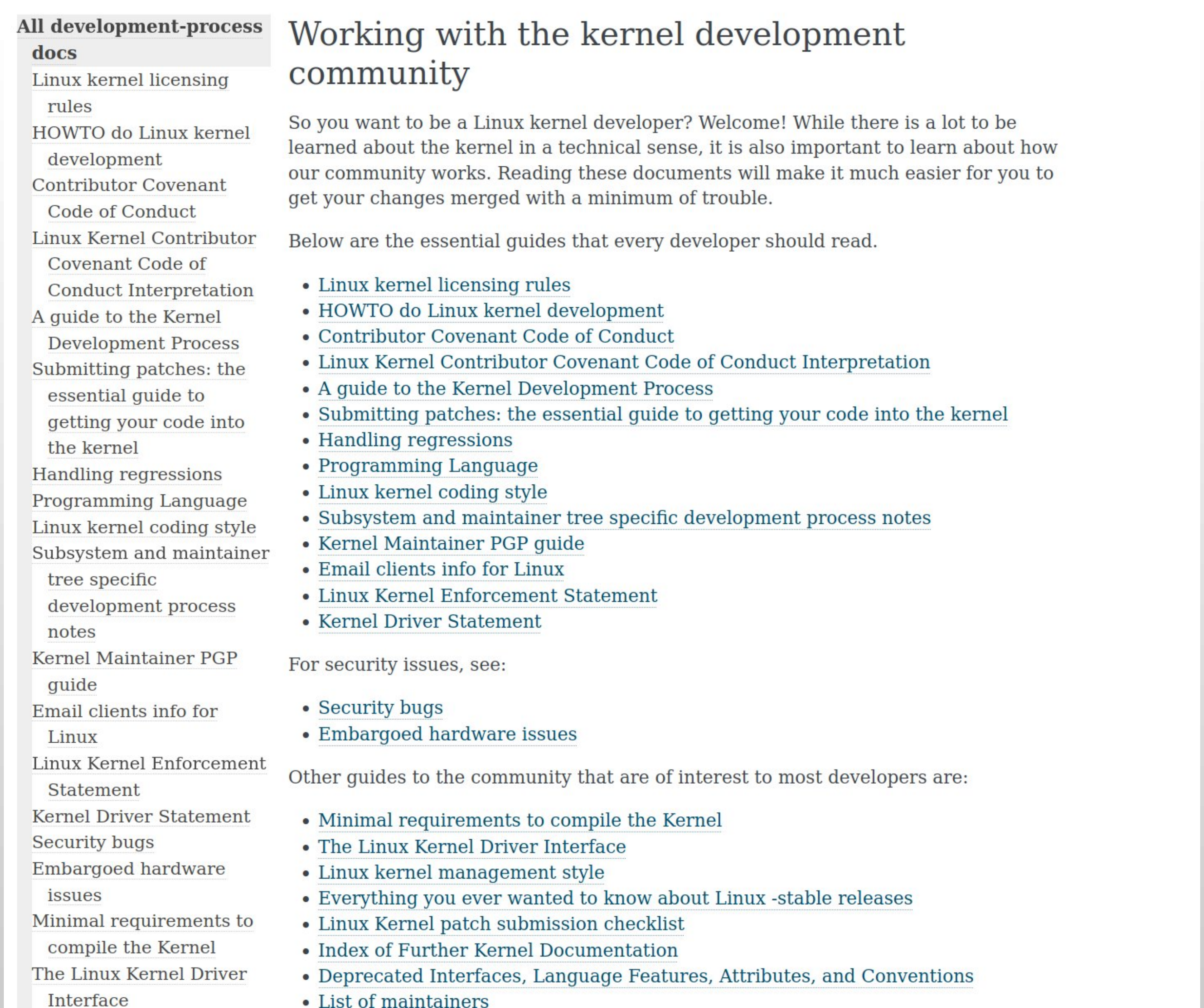
- process/coding-style
- process/submitting-patches
- process/maintainer-netdev
- admin-guide/kernel-parameters
- admin-guide/reporting-issues
- core-api/kernel-api
- driver-api/basics

Tooling

Sphinx:

- Docs-build speed
- Minimum version to support?
  - Doing 1.7 -> 2.4, go further?
- Better partial-build support
- Better sidebar

Working with the Sphinx community



Working with the kernel development community

So you want to be a Linux kernel developer? Welcome! While there is a lot to be learned about the kernel in a technical sense, it is also important to learn about how our community works. Reading these documents will make it much easier for you to get your changes merged with a minimum of trouble.

Below are the essential guides that every developer should read.

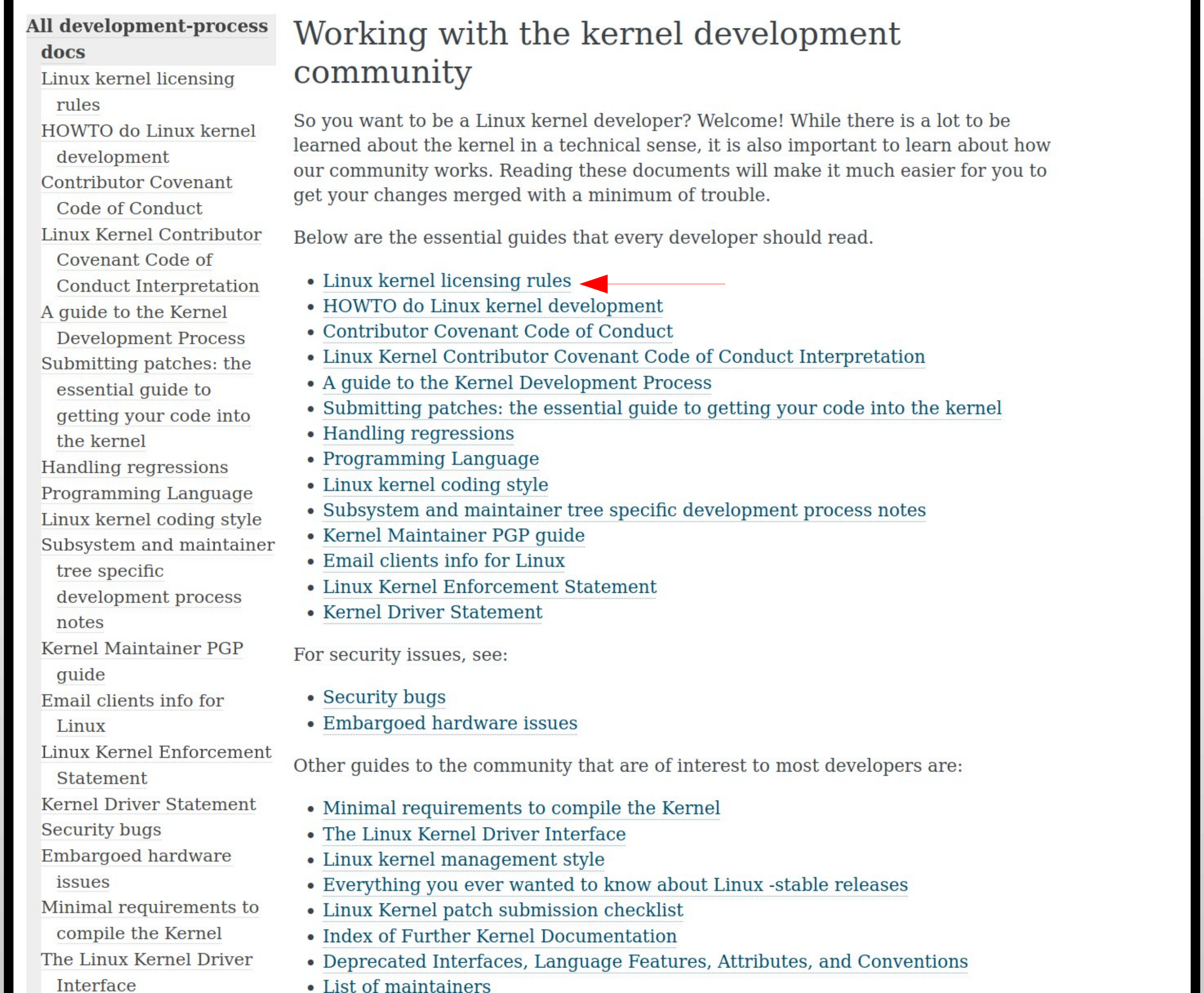
- Linux kernel licensing rules
- HOWTO do Linux kernel development
- Contributor Covenant Code of Conduct
- Linux Kernel Contributor Covenant Code of Conduct Interpretation
- A guide to the Kernel Development Process
- Submitting patches: the essential guide to getting your code into the kernel
- Handling regressions
- Programming Language
- Linux kernel coding style
- Subsystem and maintainer tree specific development process notes
- Kernel Maintainer PGP guide
- Email clients info for Linux
- Linux Kernel Enforcement Statement
- Kernel Driver Statement

For security issues, see:

- Security bugs
- Embargoed hardware issues

Other guides to the community that are of interest to most developers are:

- Minimal requirements to compile the Kernel
- The Linux Kernel Driver Interface
- Linux kernel management style
- Everything you ever wanted to know about Linux -stable releases
- Linux Kernel patch submission checklist
- Index of Further Kernel Documentation
- Deprecated Interfaces, Language Features, Attributes, and Conventions
- List of maintainers



All development-process docs

- Linux kernel licensing rules
- HOWTO do Linux kernel development
- Contributor Covenant Code of Conduct
- Linux Kernel Contributor Covenant Code of Conduct Interpretation
- A guide to the Kernel Development Process
- Submitting patches: the essential guide to getting your code into the kernel
- Handling regressions
- Programming Language
- Linux kernel coding style
- Subsystem and maintainer tree specific development process notes
- Kernel Maintainer PGP guide
- Email clients info for Linux
- Linux Kernel Enforcement Statement
- Kernel Driver Statement
- Security bugs
- Embargoed hardware issues
- Minimal requirements to compile the Kernel
- The Linux Kernel Driver Interface

Working with the kernel development community

So you want to be a Linux kernel developer? Welcome! While there is a lot to be learned about the kernel in a technical sense, it is also important to learn about how our community works. Reading these documents will make it much easier for you to get your changes merged with a minimum of trouble.

Below are the essential guides that every developer should read.

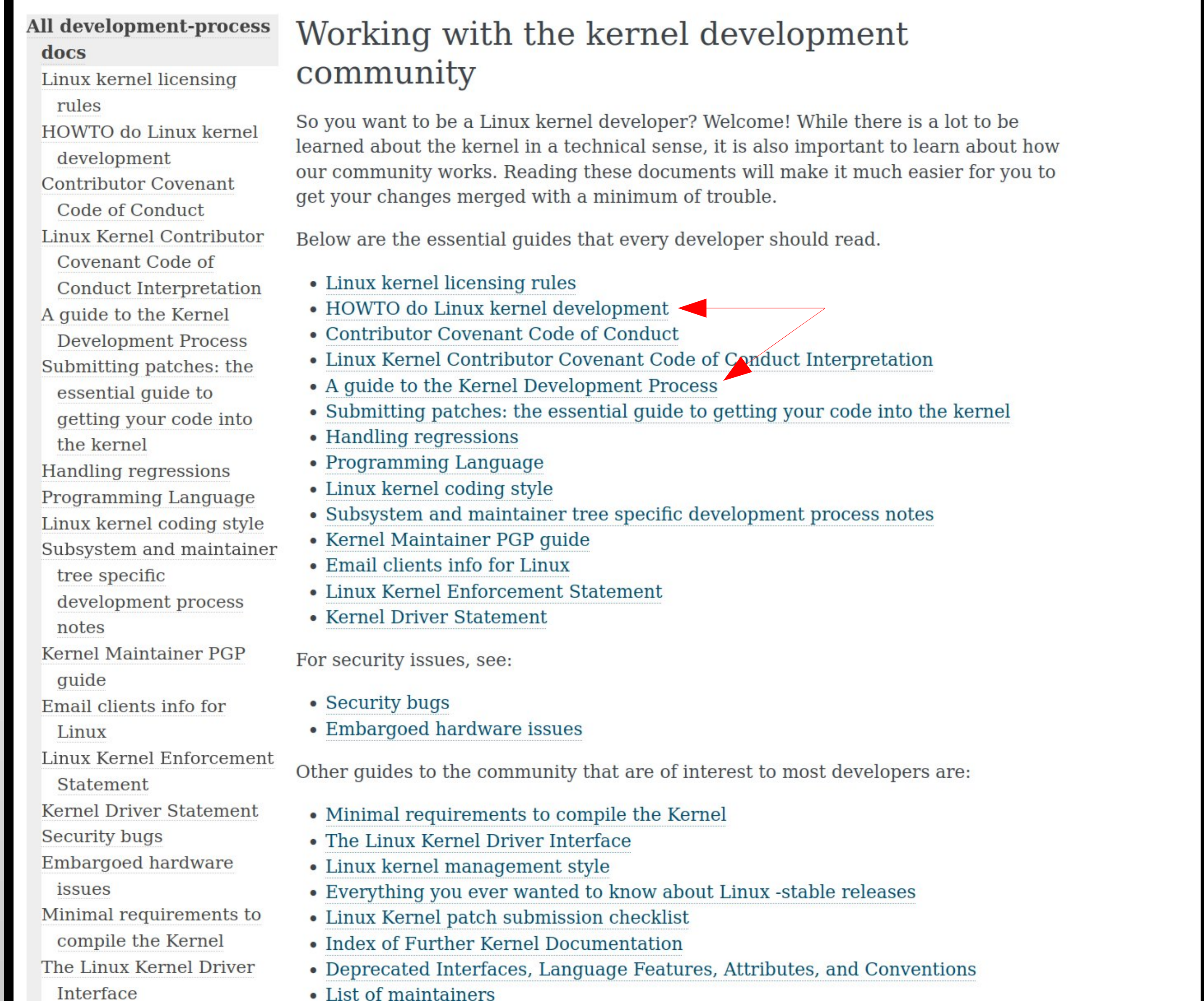
- Linux kernel licensing rules
- HOWTO do Linux kernel development
- Contributor Covenant Code of Conduct
- Linux Kernel Contributor Covenant Code of Conduct Interpretation
- A guide to the Kernel Development Process
- Submitting patches: the essential guide to getting your code into the kernel
- Handling regressions
- Programming Language
- Linux kernel coding style
- Subsystem and maintainer tree specific development process notes
- Kernel Maintainer PGP guide
- Email clients info for Linux
- Linux Kernel Enforcement Statement
- Kernel Driver Statement

For security issues, see:

- Security bugs
- Embargoed hardware issues

Other guides to the community that are of interest to most developers are:

- Minimal requirements to compile the Kernel
- The Linux Kernel Driver Interface
- Linux kernel management style
- Everything you ever wanted to know about Linux -stable releases
- Linux Kernel patch submission checklist
- Index of Further Kernel Documentation
- Deprecated Interfaces, Language Features, Attributes, and Conventions
- List of maintainers



Working with the kernel development community

So you want to be a Linux kernel developer? Welcome! While there is a lot to be learned about the kernel in a technical sense, it is also important to learn about how our community works. Reading these documents will make it much easier for you to get your changes merged with a minimum of trouble.

Below are the essential guides that every developer should read.

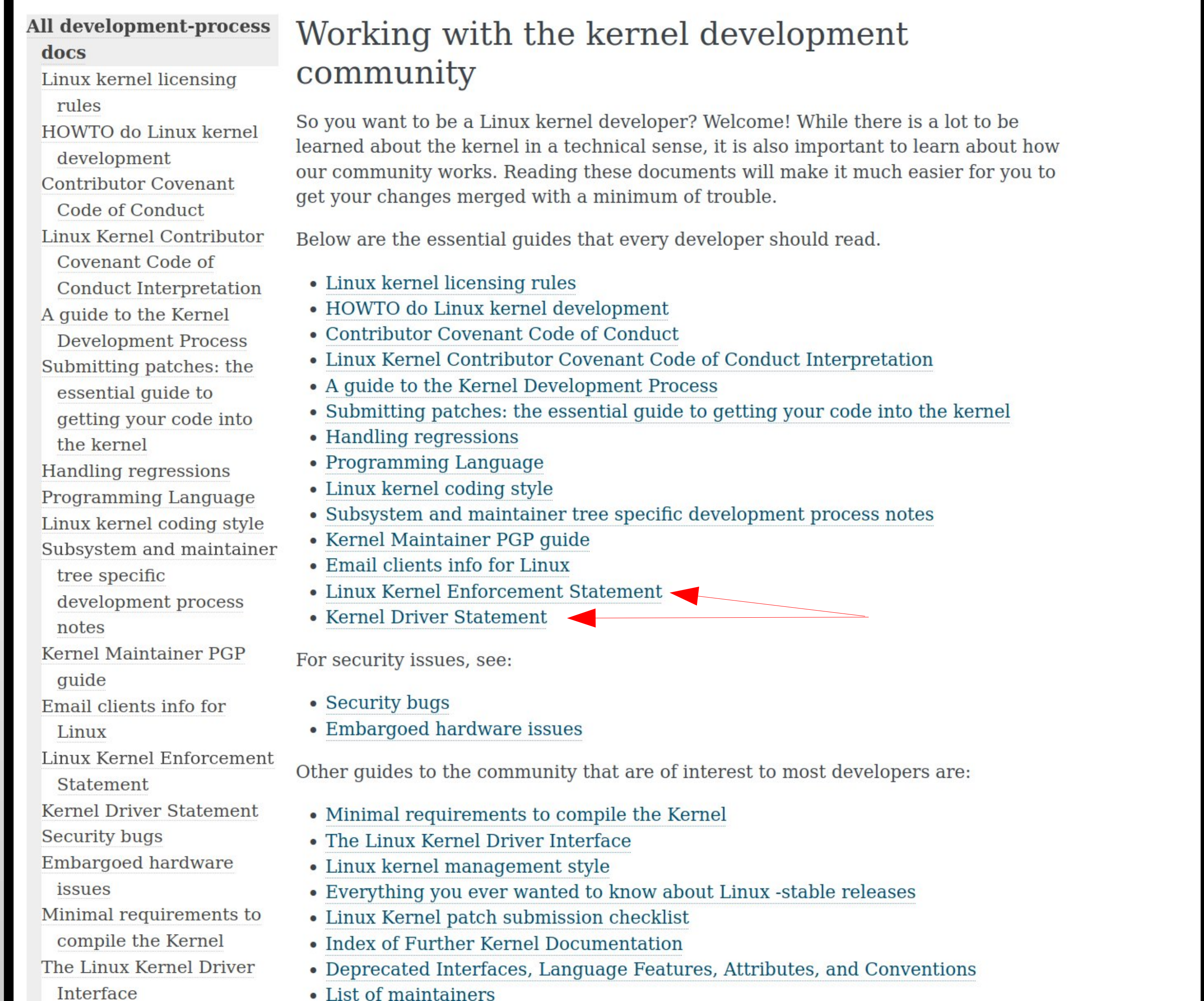
- Linux kernel licensing rules
- HOWTO do Linux kernel development
- Contributor Covenant Code of Conduct
- Linux Kernel Contributor Covenant Code of Conduct Interpretation
- A guide to the Kernel Development Process
- Submitting patches: the essential guide to getting your code into the kernel
- Handling regressions
- Programming Language
- Linux kernel coding style
- Subsystem and maintainer tree specific development process notes
- Kernel Maintainer PGP guide
- Email clients info for Linux
- Linux Kernel Enforcement Statement
- Kernel Driver Statement

For security issues, see:

- Security bugs
- Embargoed hardware issues

Other guides to the community that are of interest to most developers are:

- Minimal requirements to compile the Kernel
- The Linux Kernel Driver Interface
- Linux kernel management style
- Everything you ever wanted to know about Linux -stable releases
- Linux Kernel patch submission checklist
- Index of Further Kernel Documentation
- Deprecated Interfaces, Language Features, Attributes, and Conventions
- List of maintainers



All development-process docs

- Linux kernel licensing rules
- HOWTO do Linux kernel development
- Contributor Covenant Code of Conduct
- Linux Kernel Contributor Covenant Code of Conduct Interpretation
- A guide to the Kernel Development Process
- Submitting patches: the essential guide to getting your code into the kernel
- Handling regressions
- Programming Language
- Linux kernel coding style
- Subsystem and maintainer tree specific development process notes
- Kernel Maintainer PGP guide
- Email clients info for Linux
- Linux Kernel Enforcement Statement
- Kernel Driver Statement
- Security bugs
- Embargoed hardware issues
- Minimal requirements to compile the Kernel
- The Linux Kernel Driver Interface

Working with the kernel development community

So you want to be a Linux kernel developer? Welcome! While there is a lot to be learned about the kernel in a technical sense, it is also important to learn about how our community works. Reading these documents will make it much easier for you to get your changes merged with a minimum of trouble.

Below are the essential guides that every developer should read.

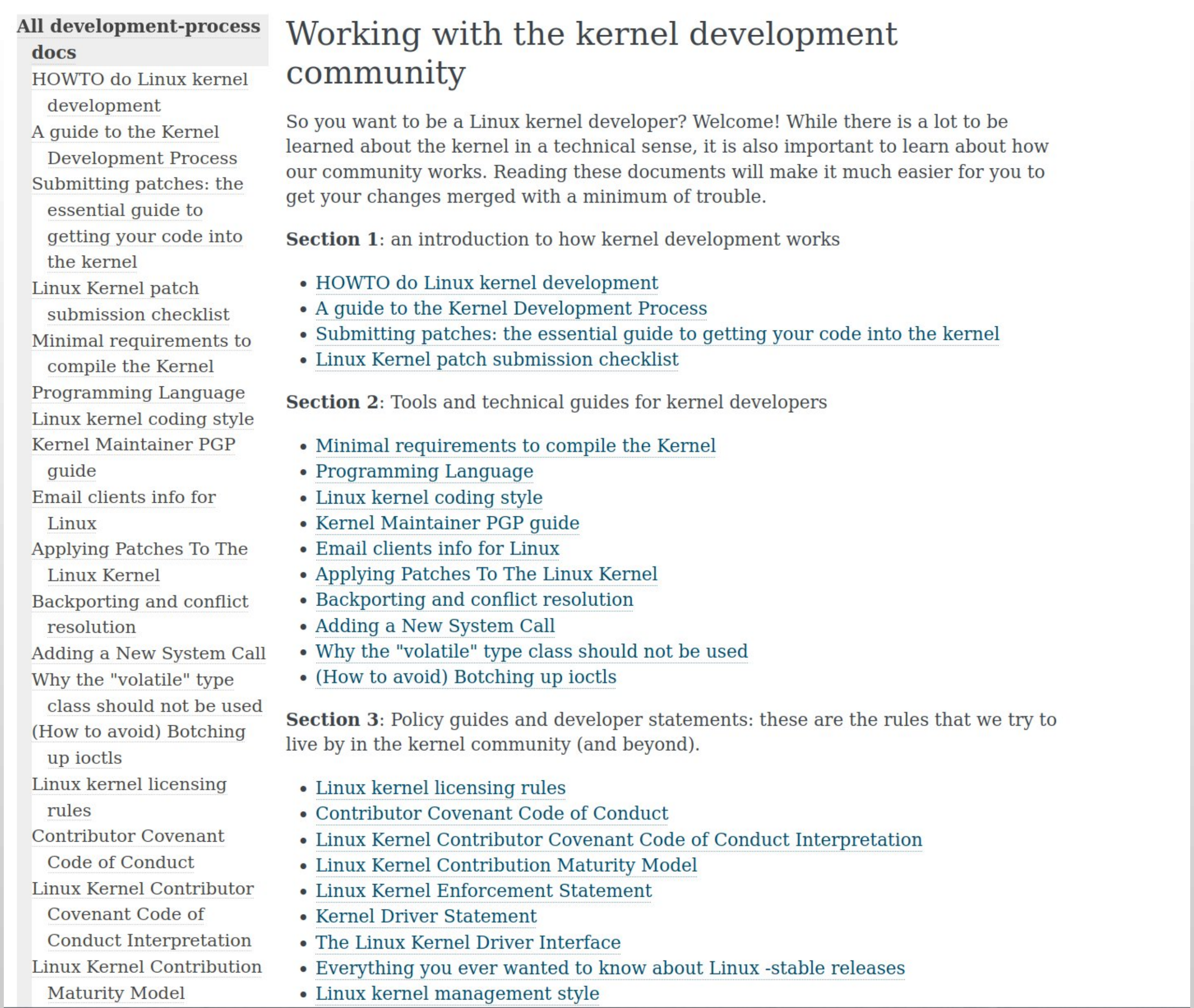
- Linux kernel licensing rules
- HOWTO do Linux kernel development
- Contributor Covenant Code of Conduct
- Linux Kernel Contributor Covenant Code of Conduct Interpretation
- A guide to the Kernel Development Process
- Submitting patches: the essential guide to getting your code into the kernel
- Handling regressions
- Programming Language
- Linux kernel coding style
- Subsystem and maintainer tree specific development process notes
- Kernel Maintainer PGP guide
- Email clients info for Linux
- Linux Kernel Enforcement Statement
- Kernel Driver Statement

For security issues, see:

- Security bugs
- Embargoed hardware issues

Other guides to the community that are of interest to most developers are:

- Minimal requirements to compile the Kernel
- The Linux Kernel Driver Interface
- Linux kernel management style
- Everything you ever wanted to know about Linux -stable releases
- Linux Kernel patch submission checklist
- Index of Further Kernel Documentation
- Deprecated Interfaces, Language Features, Attributes, and Conventions
- List of maintainers



Working with the kernel development community

So you want to be a Linux kernel developer? Welcome! While there is a lot to be learned about the kernel in a technical sense, it is also important to learn about how our community works. Reading these documents will make it much easier for you to get your changes merged with a minimum of trouble.

Section 1: an introduction to how kernel development works

- HOWTO do Linux kernel development
- A guide to the Kernel Development Process
- Submitting patches: the essential guide to getting your code into the kernel
- Linux Kernel patch submission checklist

Section 2: Tools and technical guides for kernel developers

- Minimal requirements to compile the Kernel
- Programming Language
- Linux kernel coding style
- Kernel Maintainer PGP guide
- Email clients info for Linux
- Applying Patches To The Linux Kernel
- Backporting and conflict resolution
- Adding a New System Call
- Why the "volatile" type class should not be used
- (How to avoid) Botching up ioctls

Section 3: Policy guides and developer statements: these are the rules that we try to live by in the kernel community (and beyond).

- Linux kernel licensing rules
- Contributor Covenant Code of Conduct
- Linux Kernel Contributor Covenant Code of Conduct Interpretation
- Linux Kernel Contribution Maturity Model
- Linux Kernel Enforcement Statement
- Kernel Driver Statement
- The Linux Kernel Driver Interface
- Everything you ever wanted to know about Linux -stable releases
- Linux kernel management style

How do we support this work?