GNU Taler Merchant API Tutorial

Release 0.9.0

GNU Taler team

Oct 02, 2023

CONTENTS

CHAPTER

ONE

INTRODUCTION

1.1 About GNU Taler

GNU Taler is an open protocol for an electronic payment system with a free software reference implementation. GNU Taler offers secure, fast and easy payment processing using well understood cryptographic techniques. GNU Taler allows customers to remain anonymous, while ensuring that merchants can be held accountable by governments. Hence, GNU Taler is compatible with anti-money-laundering (AML) and know-your-customer (KYC) regulation, as well as data protection regulation (such as GDPR).

1.2 About this tutorial

This tutorial addresses how to process payments using the GNU Taler merchant Backend. The audience for this tutorial are *developers* of merchants (such as Web shops) that are working on integrating GNU Taler with the customer-facing Frontend and the staff-facing Backoffice.

This chapter explains some basic concepts. In the second chapter, you will learn how to do basic payments.

This version of the tutorial has examples for Python3. It uses the `requests` library for HTTP requests. Versions for other languages/environments are available as well.

If you want to look at some simple, running examples, check out these:

- The essay merchant that sells single chapters of a book.
- The donation page that accepts donations for software projects and gives donation receipts.
- The survey that gives users who answer a question a small reward.
- The WooCommerce plugin which is a comprehensive integration into a Web shop including the refund business process.

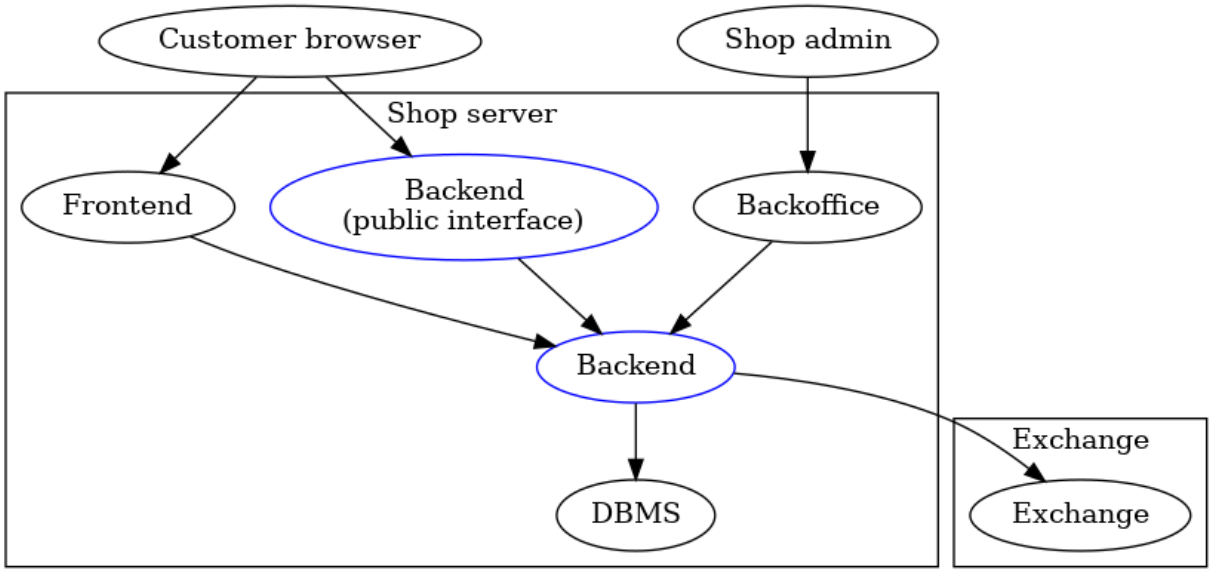
1.3 Architecture overview

The Taler software stack for a merchant consists of the following main components:

- A frontend which interacts with the customer's browser. The frontend enables the customer to build a shopping cart and place an order. Upon payment, it triggers the respective business logic to satisfy the order. This component is not included with Taler, but rather assumed to exist at the merchant. This tutorial describes how to develop a Taler frontend.

- A Taler-specific payment backend which makes it easy for the frontend to process financial transactions with Taler. For this tutorial, you will use a public sandbox backend. For production use, you must either set up your own backend or ask another person to do so for you.

The following image illustrates the various interactions of these key components:



The backend provides the cryptographic protocol support, stores Taler-specific financial information and communicates with the GNU Taler exchange over the Internet. The frontend accesses the backend via a RESTful API. As a result, the frontend never has to directly communicate with the exchange, and also does not deal with sensitive data. In particular, the merchant's signing keys and bank account information are encapsulated within the Taler backend.

Some functionality of the backend (the "public interface") is exposed to the customer's browser directly. In the HTTP API, all private endpoints (for the Backoffice) are prefixed with `/private/`. This tutorial focuses on the `/private/` endpoints. The public interface is directly used by the wallet and not relevant for the merchant (other than that the API must be exposed).

1.4 Public Sandbox Backend and Authentication

How the frontend authenticates to the Taler backend depends on the configuration. See taler-merchant-manual.

The public sandbox backend https://backend.demo.taler.net/ uses an API key in the `Authorization` header. The value of this header must be `Bearer secret-token:sandbox` for the public sandbox backend.

```
>>> import requests
>>> requests.get("https://backend.demo.taler.net/private/orders",
...              headers={"Authorization": "Bearer secret-token:sandbox"})
<Response [200]>
```

If an HTTP status code other than 200 is returned, something went wrong. You should figure out what the problem is before continuing with this tutorial.

The sandbox backend https://backend.demo.taler.net/ uses `KUDOS` as an imaginary currency. Coins denominated in `KUDOS` can be withdrawn from https://bank.demo.taler.net/.

1.5 Merchant Instances

A single Taler merchant backend server can be used by multiple merchants that are separate business entities. Each of these separate business entities is assigned a *merchant instance* which is identified by an alphanumeric *instance id*. If the instance is omitted, the instance id `default` is assumed.

The following merchant instances are configured on https://backend.demo.taler.net/:

- `GNUnet` (The GNUnet project), reachable at https://backend.demo.taler.net/instances/gnunet/
- `FSF` (The Free Software Foundation), reachable at https://backend.demo.taler.net/instances/fsf/
- `Tor` (The Tor Project), reachable at https://backend.demo.taler.net/instances/tor/
- `default` (Kudos Inc.), reachable at https://backend.demo.taler.net/

Note: These are fictional merchants used for our demonstrators and not affiliated with or officially approved by the respective projects.

All endpoints for instances offer the same API. Thus, which instance to be used is simply included in the base URL of the merchant backend.

CHAPTER

TWO

MERCHANT PAYMENT PROCESSING

2.1 Creating an Order for a Payment

Payments in Taler revolve around an *order*, which is a machine-readable description of the business transaction for which the payment is to be made. Before accepting a Taler payment as a merchant you must create such an order.

This is done by POSTing a JSON object to the backend's `/private/orders` API endpoint. At least the following fields must be given inside the `order` field:

- `amount`: The amount to be paid, as a string in the format `CURRENCY:DECIMAL_VALUE`, for example `EUR:10` for 10 Euros or `KUDOS:1.5` for 1.5 KUDOS.
- `summary`: A human-readable summary for what the payment is about. The summary should be short enough to fit into titles, though no hard limit is enforced.
- `fulfillment_url`: A URL that will be displayed once the payment is completed. For digital goods, this should be a page that displays the product that was purchased. On successful payment, the wallet automatically appends the `order_id` as a query parameter, as well as the `session_sig` for session-bound payments (discussed below).

Orders can have many more fields, see *The Taler Order Format*. When POSTing an order, you can also specify additional details such as an override for the refund duration and instructions for inventory management. These are rarely needed and not covered in this tutorial; please see the core/api-merchant reference manual for details.

A minimal Python snippet for creating an order would look like this:

```
>>> import requests
>>> body = dict(order=dict(amount="KUDOS:10",
...                        summary="Donation",
...                        fulfillment_url="https://example.com/thanks.html"),
...             create_token=False)
>>> response = requests.post("https://backend.demo.taler.net/private/orders",
...                json=body,
...                headers={"Authorization": "Bearer secret-token:sandbox"})
<Response [200]>
```

The backend will fill in some details missing in the order, such as the address of the merchant instance. The full details are called the *contract terms*.

Note: The above request disables the use of claim tokens by setting the `create_token` option to `false`. If you need claim tokens, you must adjust the code to construct the `taler://pay/` URI given below to include the claim token.

After successfully `POST`ing to `/private/orders`, a JSON with just an `order_id` field with a string representing the order ID will be returned. If you also get a claim token, please double-check that you used the request as described

above.

Together with the merchant `instance`, the order id uniquely identifies the order within a merchant backend. Using the order ID, you can trivially construct the respective `taler://pay/` URI that must be provided to the wallet. Let `example.com` be the domain name where the public endpoints of the instance are reachable. The Taler pay URI is then simply `taler://pay/example.com/$ORDER_ID/` where `$ORDER_ID` must be replaced with the ID of the order that was returned.

You can put the `taler://` URI as the target of a link to open the Taler wallet via the `taler://` schema, or put it into a QR code. However, for a Web shop, the easiest way is to simply redirect the browser to `https://example.com/orders/$ORDER_ID`. That page will then trigger the Taler wallet. Here the backend generates the right logic to trigger the wallet, supporting the various types of Taler wallets in existence. Instead of constructing the above URL by hand, it is best to obtain it by checking for the payment status as described in the next section.

2.2 Checking Payment Status and Prompting for Payment

Given the order ID, the status of a payment can be checked with the `/private/orders/$ORDER_ID` endpoint. If the payment is yet to be completed by the customer, `/private/orders/$ORDER_ID` will give the frontend a URL (under the name `payment_redirect_url`) that will trigger the customer's wallet to execute the payment. This is basically the `https://example.com/orders/$ORDER_ID` URL we discussed above.

Note that the best way to obtain the `payment_redirect_url` is to check the status of the payment, even if you know that the user did not pay yet. There are a few corner cases to consider when constructing this URL, so asking the backend to do it is the safest method.

```
>>> import requests
>>> r = requests.get("https://backend.demo.taler.net/private/orders/" + order_id,
...                  headers={"Authorization": "Bearer secret-token:sandbox"})
>>> print(r.json())
```

If the `order_status` field in the response is `paid`, you will not get a `payment_redirect_url` and instead information about the payment status, including:

- `contract_terms`: The full contract terms of the order.
- `refunded`: `true` if a (possibly partial) refund was granted for this purchase.
- `refunded_amount`: Amount that was refunded

Once the frontend has confirmed that the payment was successful, it usually needs to trigger the business logic for the merchant to fulfill the merchant's obligations under the contract.

Note: You do not need to keep querying to notice changes to the order's transaction status. The endpoints support long polling, simply specify a `timeout_ms` query parameter with how long you want to wait at most for the order status to change to `paid`.

CHAPTER

THREE

GIVING REFUNDS

A refund in GNU Taler is a way to "undo" a payment. It needs to be authorized by the merchant. Refunds can be for any fraction of the original amount paid, but they cannot exceed the original payment. Refunds are time-limited and can only happen while the exchange holds funds for a particular payment in escrow. The time during which a refund is possible can be controlled by setting the `refund_deadline` in an order. The default value for this refund deadline is specified in the configuration of the merchant's backend.

The frontend can instruct the merchant backend to authorize a refund by `POST`ing to the `/private/orders/$ORDER_ID/refund` endpoint.

The refund request JSON object has only two fields:

- `refund`: Amount to be refunded. If a previous refund was authorized for the same order, the new amount must be higher, otherwise the operation has no effect. The value indicates the total amount to be refunded, *not* an increase in the refund.
- `reason`: Human-readable justification for the refund. The reason is only used by the Back Office and is not exposed to the customer.

If the request is successful (indicated by HTTP status code 200), the response includes a `taler_refund_uri`. The frontend must redirect the customer's browser to that URL to allow the refund to be processed by the wallet.

This code snipped illustrates giving a refund:

```
>>> import requests
>>> refund_req = dict(refund="KUDOS:10",
...                   reason="Customer did not like the product")
>>> requests.post("https://backend.demo.taler.net/private/orders/"
...               + order_id + "/refund", json=refund_req,
...               headers={"Authorization": "Bearer secret-token:sandbox"})
<Response [200]>
```

Note: After granting a refund, the public `https://example.com/orders/$ORDER_ID` endpoint will change its wallet interaction from requesting payment to offering a refund. Thus, frontends may again redirect browsers to this endpoint. However, to do so, a `h_contract` field must be appended (`?h_contract=$H_CONTRACT`) as the public endpoint requires it to authenticate the client. The required `$H_CONTRACT` value is returned in the refund response under the `h_contract` field.

CHAPTER

FOUR

REPURCHASE DETECTION AND FULFILLMENT URLS

A possible problem for merchants selling access to digital articles is that a customer may have paid for an article on one device, but may then want to read it on a different device, possibly one that does not even have a Taler wallet installed.

Naturally, at this point the customer would at first still be prompted to pay for the article again. If the customer then opens the `taler://` link in the wallet that did previously pay for the article (for example by scanning the QR code on the desktop with the Android App), the wallet will claim the contract, detect that the fulfillment URL is identical to one that it already has made a payment for in the past, and initiate **repurchase redirection**: Here, the wallet will contact the merchant and replay the previous payment, except this time using the (current) session ID of the browser (it learns the session ID from the QR code).

The merchant backend then updates the session ID of the existing order to the current session ID of the browser. When the payment status for the "new" unpaid order is checked (or already in long-polling), the backend detects that for the browser's *session ID* and *fulfillment URL* there is an existing paid contract. It then tells the browser to immediately redirect to the fulfillment URL where the already paid article is available.

To ensure this mechanism works as designed, merchants must make sure to not use the same fulfillment URL for different products or for physical products where customers may be expected to buy the article repeatedly. Similarly, it is crucial that merchants consistently use the same fulfillment URL for the same digital product where repurchase detection is desired.

Note that changing the session ID to a different device requires the involvement of the wallet that made the payment, thus reasonably limiting the possibility of broadly sharing the digital purchases. Repurchase detection is also *only* done for HTTP(S) fulfillment URLs. In particular, this means fulfillment URIs like `taler://fulfillment-success/$MESSAGE` are not considered to identify a resource you can pay for and thus do not have to be unique.

CHAPTER

FIVE

GIVING CUSTOMERS REWARDS

GNU Taler allows Web sites to grant digital cash directly to a visitor. The idea is that some sites may want incentivize actions such as filling out a survey or trying a new feature. It is important to note that receiving rewards is not enforceable for the visitor, as there is no contract. It is simply a voluntary gesture of appreciation of the site to its visitor. However, once a reward has been granted, the visitor obtains full control over the funds provided by the site.

The merchant backend of the site must be properly configured for rewards, and sufficient funds must be made available for rewards. See the Taler User Guide for details.

To check if rewards are configured properly and if there are sufficient funds available for granting rewards, query the `/private/reserves` endpoint:

```
>>> import requests
>>> requests.get("https://backend.demo.taler.net/private/reserves",
...              headers={"Authorization": "Bearer secret-token:sandbox"})
<Response [200]>
```

Check that a reserve exists where the `merchant_initial_amount` is below the `committed_amount` and that the reserve is `active`.

To authorize a reward, `POST` to `/private/rewards`. The following fields are recognized in the JSON request object:

- `amount`: Amount that should be given to the visitor as a reward.
- `justification`: Description of why the reward was granted. Human-readable text not exposed to the customer, but used by the Back Office.
- `next_url`: The URL that the user's browser should be redirected to by the wallet, once the reward has been processed.

The response from the backend contains a `taler_reward_url`. The customer's browser must be redirected to this URL for the wallet to pick up the reward.

This code snipped illustrates giving a reward:

```
>>> import requests
>>> reward_req = dict(amount="KUDOS:0.5",
...                   justification="User filled out survey",
...                   next_url="https://merchant.com/thanks.html")
>>> requests.post("https://backend.demo.taler.net/private/rewards", json=reward_req,
...              headers={"Authorization": "Bearer secret-token:sandbox"})
<Response [200]>
```

CHAPTER

SIX

ADVANCED TOPICS

6.1 Session-Bound Payments

Sometimes checking if an order has been paid for is not enough. For example, when selling access to online media, the publisher may want to be paid for exactly the same product by each customer. Taler supports this model by allowing the mechant to check whether the "payment receipt" is available on the user's current device. This prevents users from easily sharing media access by transmitting a link to the fulfillment page. Of course, sophisticated users could share payment receipts as well, but this is not as easy as sharing a link, and in this case they are more likely to just share the media directly.

To use this feature, the merchant must first assign the user's current browser an ephemeral `session_id`, usually via a session cookie. When executing or re-playing a payment, the wallet will receive an additional signature (`session_sig`). This signature certifies that the wallet showed a payment receipt for the respective order in the current session.

Session-bound payments are triggered by passing the `session_id` parameter to the `/check-payment` endpoint. The wallet will then redirect to the fulfillment page, but include an additional `session_sig` parameter. The frontend can query `/check-payment` with both the `session_id` and the `session_sig` to verify that the signature is correct.

The last session ID that was successfully used to prove that the payment receipt is in the user's wallet is also available as `last_session_id` in the response to `/check-payment`.

6.2 Product Identification

In some situations the user may have paid for some digital good, but the frontend does not know the exact order ID, and thus cannot instruct the wallet to reveal the existing payment receipt. This is common for simple shops without a login system. In this case, the user would be prompted for payment again, even though they already purchased the product.

To allow the wallet to instead find the existing payment receipt, the shop must use a unique fulfillment URL for each product. Then, the frontend must provide an additional `resource_url` parameter to to `/check-payment`. It should identify this unique fulfillment URL for the product. The wallet will then check whether it has paid for a contract with the same `resource_url` before, and if so replay the previous payment.

6.3 The Taler Order Format

A Taler order can specify many details about the payment. This section describes each of the fields in depth.

Financial amounts are always specified as a string in the format `"CURRENCY:DECIMAL_VALUE"`.

amount
: Specifies the total amount to be paid to the merchant by the customer.

max_fee
: This is the maximum total amount of deposit fees that the merchant is willing to pay. If the deposit fees for the coins exceed this amount, the customer has to include it in the payment total. The fee is specified using the same format used for `amount`.

max_wire_fee
: Maximum wire fee accepted by the merchant (customer share to be divided by the `wire_fee_amortization` factor, and further reduced if deposit fees are below `max_fee`). Default if missing is zero.

wire_fee_amortization
: Over how many customer transactions does the merchant expect to amortize wire fees on average? If the exchange's wire fee is above `max_wire_fee`, the difference is divided by this number to compute the expected customer's contribution to the wire fee. The customer's contribution may further be reduced by the difference between the `max_fee` and the sum of the actual deposit fees. Optional, default value if missing is 1. Zero and negative values are invalid and also interpreted as 1.

pay_url
: Which URL accepts payments. This is the URL where the wallet will POST coins.

fulfillment_url
: Which URL should the wallet go to for obtaining the fulfillment, for example the HTML or PDF of an article that was bought, or an order tracking system for shipments, or a simple human-readable Web page indicating the status of the contract.

order_id
: Alphanumeric identifier, freely definable by the merchant. Used by the merchant to uniquely identify the transaction.

summary
: Short, human-readable summary of the contract. To be used when displaying the contract in just one line, for example in the transaction history of the customer.

timestamp
: Time at which the offer was generated.

pay_deadline
: Timestamp of the time by which the merchant wants the exchange to definitively wire the money due from this contract. Once this deadline expires, the exchange will aggregate all deposits where the contracts are past the `refund_deadline` and execute one large wire payment for them. Amounts will be rounded down to the wire transfer unit; if the total amount is still below the wire transfer unit, it will not be disbursed.

refund_deadline
: Timestamp until which the merchant willing (and able) to give refunds for the contract using Taler. Note that the Taler exchange will hold the payment in escrow at least until this deadline. Until this time, the merchant will be able to sign a message to trigger a refund to the customer. After this time, it will no longer be possible to refund the customer. Must be smaller than the `pay_deadline`.

products
: Array of products that are being sold to the customer. Each entry contains a tuple with the following values:

description
Description of the product.

quantity
Quantity of the items to be shipped. May specify a unit (e.g. `1 kg`) or just the count.

price
Price for `quantity` units of this product shipped to the given `delivery_location`. Note that usually the sum of all of the prices should add up to the total amount of the contract, but it may be different due to discounts or because individual prices are unavailable.

product_id
Unique ID of the product in the merchant's catalog. Can generally be chosen freely as it only has meaning for the merchant, but should be a number in the range $[0, 2^{51})$.

taxes
Map of applicable taxes to be paid by the merchant. The label is the name of the tax, i.e. VAT, sales tax or income tax, and the value is the applicable tax amount. Note that arbitrary labels are permitted, as long as they are used to identify the applicable tax regime. Details may be specified by the regulator. This is used to declare to the customer which taxes the merchant intends to pay, and can be used by the customer as a receipt. The information is also likely to be used by tax audits of the merchant.

delivery_date
Time by which the product is to be delivered to the `delivery_location`.

delivery_location
This should give a label in the `locations` map, specifying where the item is to be delivered.

Values can be omitted if they are not applicable. For example, if a purchase is about a bundle of products that have no individual prices or product IDs, the `product_id` or `price` may not be specified in the contract. Similarly, for virtual products delivered directly via the fulfillment URI, there is no `delivery_location`.

merchant

address
This should give a label in the `locations` map, specifying where the merchant is located.

name
This should give a human-readable name for the merchant's business.

jurisdiction
This should give a label in the `locations` map, specifying the jurisdiction under which this contract is to be arbitrated.

locations
Associative map of locations used in the contract. Labels for locations in this map can be freely chosen and used whenever a location is required in other parts of the contract. This way, if the same location is required many times (such as the business address of the customer or the merchant), it only needs to be listed (and transmitted) once, and can otherwise be referred to via the label. A non-exhaustive list of location attributes is the following:

name
Receiver name for delivery, either business or person name.

country
Name of the country for delivery, as found on a postal package, e.g. "France".

state
Name of the state for delivery, as found on a postal package, e.g. "NY".

region
Name of the region for delivery, as found on a postal package.

province
Name of the province for delivery, as found on a postal package.

city
Name of the city for delivery, as found on a postal package.

zip_code
ZIP code for delivery, as found on a postal package.

street
Street name for delivery, as found on a postal package.

street_number
Street number (number of the house) for delivery, as found on a postal package.

Note: Locations are not required to specify all of these fields, and they is also allowed to have additional fields. Contract renderers must render at least the fields listed above, and should render fields that they do not understand as a key-value list.

APPENDIX

A

GNU FREE DOCUMENTATION LICENSE

Version 1.3, 3 November 2008

A.1 0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other functional and useful document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU Affero General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

A.2 1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The “Invariant Sections” are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The “Cover Texts” are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A “Transparent” copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not “Transparent” is called “Opaque”.

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The “Title Page” means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, “Title Page” means the text near the most prominent appearance of the work’s title, preceding the beginning of the body of the text.

The “publisher” means any person or entity that distributes copies of the Document to the public.

A section “Entitled XYZ” means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as “Acknowledgements”, “Dedications”, “Endorsements”, or “History”.) To “Preserve the Title” of such a section when you modify the Document means that it remains a section “Entitled XYZ” according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

A.3 2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

A.4 3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

A.5 4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

- A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.
- B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.
- C. State on the Title page the name of the publisher of the Modified Version, as the publisher.
- D. Preserve all the copyright notices of the Document.
- E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.
- F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.
- G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.
- H. Include an unaltered copy of this License.
- I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.

- J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.
- K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.
- L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.
- M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.
- N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.
- O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various partiesâ€"for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

A.6 5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements".

A.7 6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

A.8 7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

A.9 8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

A.10 9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, or distribute it is void, and will automatically terminate your rights under this License.

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, receipt of a copy of some or all of the same material does not give you any rights to use it.

A.11 10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See https://www.gnu.org/licenses/.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation. If the Document specifies that a proxy can decide which future versions of this License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Document.

A.12 11. RELICENSING

"Massive Multiauthor Collaboration Site" (or "MMC Site") means any World Wide Web server that publishes copyrightable works and also provides prominent facilities for anybody to edit those works. A public wiki that anybody can edit is an example of such a server. A "Massive Multiauthor Collaboration" (or "MMC") contained in the site means any set of copyrightable works thus published on the MMC site.

"CC-BY-SA" means the Creative Commons Attribution-Share Alike 3.0 license published by Creative Commons Corporation, a not-for-profit corporation with a principal place of business in San Francisco, California, as well as future copyleft versions of that license published by that same organization.

"Incorporate" means to publish or republish a Document, in whole or in part, as part of another Document.

An MMC is "eligible for relicensing" if it is licensed under this License, and if all works that were first published under this License somewhere other than this MMC, and subsequently incorporated in whole or in part into the MMC, (1) had no cover texts or invariant sections, and (2) were thus incorporated prior to November 1, 2008.

The operator of an MMC Site may republish an MMC contained in the site under CC-BY-SA on the same site at any time before August 1, 2009, provided the MMC is eligible for relicensing.

A.13 ADDENDUM: How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

```
Copyright (C)  YEAR  YOUR NAME.
Permission is granted to copy, distribute and/or modify this document
under the terms of the GNU Free Documentation License, Version 1.3
or any later version published by the Free Software Foundation;
with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts.
A copy of the license is included in the section entitled "GNU
Free Documentation License".
```

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the "with â€¦ Texts." line with this:

```
with the Invariant Sections being LIST THEIR TITLES, with the
Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST.
```

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU Affero General Public License, to permit their use in free software.

INDEX