systemdタイマの操作

概要

定期的なバックアップスクリプトの実行から、コンピュータの起動後直ちに特定のプロセスを開始する処理まで、Linuxシステム上でスケジュール設定を必要とする多数のタスクがあります。systemdタイマは、ジョブやサービスをスケジュール設定し、管理するための柔軟なメカニズムを提供します。

目的

この記事は、タイマの作成、保守、テスト、トラブルシューティング、およびcronからの移行を網羅したsystemdタイマの全容の概要を提供することを目的としています。

所要時間

systemdタイマの例を作成するには10分を要します。また、systemdタイマの仕組みを十分に理解するには最長で30分が必要です。

要件

- systemdの基本的な理解。
- root特権またはsudo特権。通常のユーザとしてsystemdタイマを使用するには、まず7項 「通常のユーザとしてのタイマの使用」を参照してください。

発行日: 11/12/2025

目次

1 systemdタイマのコンセプト

systemdタイマユニットは、Linux上でジョブのスケジュールを設定する機能を提供します。これらのジョブの実行時間は、日時またはイベントに基づいて設定できます。

systemdタイマユニットは、ファイル名拡張子.timerで識別できます。各タイマファイルには、その制御対象になるサービスファイルが必要です。つまり、タイマファイルは、対応するサービスファイルを有効にし、管理します。systemdタイマは次の機能をサポートしています。

- タイマユニットを使用してスケジュールするジョブを、他のsystemdサービスに依存して動作するものとすることができます。タイマユニットは通常のsystemdサービスとして扱われるので、**systemctl**で管理できます。
- タイマは、カレンダイベントでトリガされるリアルタイムタイマまたは特定の開始点から指定された時間が経過するとトリガされる単調タイマとすることができます。
- タイマユニットはシステムジャーナルに記録されるため、監視やトラブルシューティングが容易になります。
- タイマでは、中央管理サービスであるsystemdを使用します。
- 予定のタイマ実行時間でシステムがオフになっている場合は、システムが再度稼働したときにタイマが実行されます。

2 タイマの作成

次の例では、ブート時間後にhelloworld.shシェルスクリプトをトリガし、有効化された時刻を基準にして24時間ごとに実行を繰り返すタイマを設定する方法を示しています。また、このトリガを月曜日から金曜日の毎日午前10時にも実行します。

2.1 Hello Worldの例

1. 次の内容で/etc/systemd/system/helloworld.serviceファイルを作成します。

```
[Unit]
Description="Hello World script"
```

```
[Service]
ExecStart=/usr/local/bin/helloworld.sh
```

これは、どのアプリケーションを実行するかを`systemd`に指示する`systemd`サービスファイルです。

2. 次の内容で`/etc/systemd/system/helloworld.timer`ファイルを作成します。

```
[Unit]
Description="Run helloworld.service 5min after boot and every 24 hours relative to
 activation time"

[Timer]
OnBootSec=5min
OnUnitActiveSec=24h
OnCalendar=Mon..Fri *-*-* 10:00:*
Unit=helloworld.service

[Install]
WantedBy=multi-user.target
```

これは、各サービスファイルのアクティベーションを制御するタイマファイルです。

3. 前記で作成したファイルにエラーがないことを確認します。

```
> systemd-analyze verify /etc/systemd/system/helloworld.*
```

このコマンドから出力が返されなければ、ファイルは検証に適合しています。

4. タイマを起動します。

```
> sudo systemctl start helloworld.timer
```

現在のセッションでのみタイマを有効にします。

5. タイマを有効にして、ブート時にタイマが有効になることを確認します。

```
> sudo systemctl enable helloworld.timer
```

2.2 説明に沿った例

例 1: サービスファイル

```
[Unit]
Description="Hello World script"(1)

[Service]
```

```
ExecStart=/usr/local/bin/helloworld.sh❷
```

❶ サービスファイルの目的の簡単な説明。

❷ 実行するアプリケーション。

`[Unit]`セクションと`[Service]`セクションは、サービスファイルが機能するために最小限必要なセクションです。通常、`systemd`サービスファイルには、サービスでロードする1つ以上のターゲットを指定した`[Install]`セクションがあります。この情報はタイマファイルから提供されるので、タイマのサービスファイルにこのセクションは不要です。高度な設定については Managing `systemd` targets with systemctl (https://documentation.suse.com/smart/systems-management/html/reference-managing-systemd-targets-systemctl/reference-systemctl-managing-targets.html) を参照してください。

例 2: タイマファイル

```
[Unit]
Description="Run helloworld.service 5min after boot and every 24 hours relative to
 activation time"❶

[Timer]
OnBootSec=5min❷
OnUnitActiveSec=24h❸
OnCalendar=Mon..Fri *-*-* 10:00:*❹
Unit=helloworld.service❺

[Install]
WantedBy=multi-user.target❻
```

❶ タイマファイルの簡単な説明。

❷ システムブートの5分後にサービスをトリガするタイマを指定します。詳細については単調タイマを参照してください。

❸ サービスを有効にした時点から24時間後にサービスをトリガするタイマを指定します(つまり、このタイマは1日に1回サービスをトリガします)。詳細についてはリアルタイムタイマを参照してください。

❹ 決まった時点(この例では、月曜日から金曜日の毎日午前10時)にサービスをトリガするタイマを指定します。詳細についてはリアルタイムタイマを参照してください。

❺ 実行するサービスファイル。

❻ タイマが動作する`systemd`ターゲット。`systemd`ターゲットの詳細については、Managing `systemd` targets with systemctl (https://documentation.suse.com/smart/systems-management/html/reference-managing-systemd-targets-systemctl/reference-systemctl-managing-targets.html) を参照してください。

3 タイマの管理

systemctlコマンドを使用してタイマを管理できます。

タイマの起動と停止

```
> sudo systemctl start TIMER.timer
> sudo systemctl restart TIMER.timer
> sudo systemctl stop TIMER.timer
```

タイマの有効化と無効化

```
> sudo systemctl enable TIMER.timer
> sudo systemctl disable TIMER.timer
```

タイマファイルの内容の表示

```
> sudo systemctl cat TIMER.timer
```

特定のタイマの確認

```
> sudo systemctl status TIMER.timer
```

例 3: タイマのステータス

```
> sudo systemctl status helloworld.timer
● helloworld.timer - "Run helloworld.service 5min after boot and every 24 hours
relative to activation time" ❶
Loaded: loaded (/etc/systemd/system/helloworld.timer; disabled; vendor preset:
 disabled) ❷
Active: active (waiting) since Tue 2022-10-26 18:35:41 CEST; 6s ago ❸
Trigger: Wed 2022-10-27 18:35:41 CEST; 23h left ❹
Triggers: ● helloworld.service ❺
❻
Oct 26 18:35:41 neo systemd[1]: Started "Run helloworld.service 5min after boot and
 every 24 hours relative to activation time". ❼
```

❶ タイマのファイル名と説明。

❷ タイマが正常に解析されてメモリ内に保持(ロード)されているかどうかを一覧にします。また、タイマファイルのフルパスを示し、タイマがブート時に開始されるか(有効)、されないか(無効)を示します。最初の値は現在のシステム設定を示し、2番目の値はベンダによるプリセット値を示しています。

❸ タイマがアクティブ(イベントのトリガを待機している状態)か非アクティブかを示します。アクティブな場合、前回のアクティベーションからの経過時間も表示されます(この例では6秒)。

❹ タイマが次回トリガされる日時。

❺ タイマによってトリガされるサービスファイルの名前。

❻ マニュアルページなどのドキュメントを示すオプション行。そのような情報がない場合は、この例のように空の行になります。

❼ タイマによって作成された最新のジャーナルエントリ。

システムで利用できるすべてのタイマを一覧表示するには**`systemctl list-timers`**を使用します。次のオプションを指定できます。

すべてのアクティブなタイマを一覧表示します。

```
> sudo systemctl list-timers
```

非アクティブなタイマも含め、すべてのタイマを一覧表示します。

```
> sudo systemctl list-timers --all
```

パターンに一致するすべてのタイマを一覧表示します。

```
> sudo systemctl list-timers PATTERN
> sudo systemctl list-timers --allPATTERN
```

`PATTERN`は、名前またはシェルのグロブ表現とする必要があります。オペレータ`*`、`?`、および`[]`を使用できます。グロブパターンの詳細については**`man 7 glob`**を参照してください。

特定の状態にあるタイマを一覧表示します。

```
> sudo systemctl list-timers --state=STATE
```

`STATE`に指定できる値は、`active`、`failed`、`load`、`sub`です。詳細については**`man systemctl`**を参照してください。

例 4: タイマの一覧表示

どの**`systemctl list-timers`**を実行しても次のようなテーブルが表示されます。この例では、パターン`snapper*`に一致するすべてのアクティブなタイマが一覧表示されます。

```
> sudo systemctl list-timers snapper*
NEXT❶                         LEFT❷       LAST❸                          PASSED❹
 UNIT❺                   ACTIVATES❻

------------------------------------------------------------------------------------------
```

```
Tue 2022-10-26 19:00:00 CEST 39min left Tue 2022-10-26 18:00:29 CEST 19min ago
 snapper-timeline.timer snapper-timeline.service
Wed 2022-10-27 08:33:04 CEST 14h   left Tue 2022-10-26 08:33:04 CEST 9h ago
 snapper-cleanup.timer  snapper-cleanup.service
```

❶ タイマの次回実行日時。
❷ タイマの次回実行までの残り時間。
❸ タイマの前回実行日時。
❹ タイマの前回実行からの経過時間。
❺ タイマユニットの名前。
❻ タイマで有効になるサービスの名前。

4 タイマのタイプ

systemdは、リアルタイム(カレンダに基づく)と単調(イベントに基づく)の2種類のタイマをサポートしています。一般的にタイマは永続的ですが、systemdを使用すると、現在のセッションでのみ有効な過渡タイマも設定できます。

リアルタイムタイマ

リアルタイムタイマは、カレンダイベントでトリガされます。その定義にはOnCalendarオプションを使用します。
日時に基づいてイベントをトリガするタイミングを指定できます。次のテンプレートを使用します。

```
OnCalendar=DayOfWeek❶ Year-Month-Day❷ Hour:Minute:Second❸
```

❶ 曜日。指定できる値は、Sun、Mon、Tue、Wed、Thu、Fri、Satです。曜日を無視する場合は省略します。
❷ 日付。月と日を2桁で指定し、年を4桁で指定します。該当するあらゆる日時と一致するように、各値をワイルドカード*で置き換えることができます。
❸ 時刻。各値を2桁で指定します。該当するあらゆる日時と一致するように、各値をワイルドカード*で置き換えることができます。

これらのすべての値については、2つのドットを使用して連続する範囲を定義でき(Mon..Fri)、個々の値をコンマで区切って複数の値を列挙できます(Mon,Wed,Fri)。

例 5: リアルタイムタイマの例

- 毎週金曜日の午後6時:

```
OnCalendar=Fri *-*-* 18:00:00
```

- 毎日午前5時:

```
OnCalendar=Mon..Sun *-*-* 5:00:00
```

- 日曜日と火曜日の午前1時と午前3時:

```
OnCalendar=Tue,Sun *-*-* 01,03:00:00
```

- 単一の日付:

```
OnCalendar=Mo..Sun 2023-09-23 00:00:01
```

- 時刻が異なる複数のトリガを指定するには、1つのタイマファイルに複数のOnCalendarエントリを作成します。

```
OnCalendar=Mon..Fri *-*-* 10:00
OnCalendar=Sat,Sun *-*-* 22:00
```

使用可能な機能とオプションの全リストについては、**`man 7 systemd.time`**を参照してください。ここでは、次のトピックに関する追加情報が提供されています。

- 構文の短縮と略語の使用
- 繰り返しの指定
- 月の特定の日を検索(月の最終日、最後の日曜日など)
- タイムゾーンの適用

単調タイマ

単調タイマは、システムブートやシステムユニットのアクティベーションイベントなど、特定のイベントから指定の時間が経過するとトリガされます。この値は時間の単位(分、時間、日、月、年など)で定義します。使用できる単位は、`usec`、`msec`、`seconds`、`minutes`、`hours`、`days`、`weeks`、`months`、`years`です。単調タイマの定義で使用するオプションがいくつかあります。

- `OnActiveSec`: ユニットのアクティベーションからの時間

```
OnActiveSec=50minutes
```

- `OnBootSec`: システムのブートからの時間

```
OnBootSec=10hours
```

- `OnStartupSec`: サービスマネージャの起動からの時間。システムサービスの場合、これは`OnActiveSec`にほぼ等しくなります。ユーザがログインするとサービスマネージャが起動するユーザサービスで、このオプションを使用します。

```
OnStartupSec=5minutes 20seconds
```

- `OnUnitActiveSec`: 該当サービスの前回有効化からの時間

```
OnUnitActiveSec=10seconds
```

- `OnUnitInactiveSec`: 該当サービスの前回無効化からの時間

```
OnUnitInactiveSec=2hours 15minutes 18 seconds
```

過渡タイマ

過渡タイマは、現在のセッションでのみ有効な一時的タイマです。このタイマを使用すると、既存のサービスファイルを使用するか、プログラムを直接開始できます。**`systemd-run`**を実行することで過渡タイマを呼び出します。
次の例では、2時間ごとに`helloworld.service`ユニットを実行します。

```
> sudo systemd-run --on-active="2hours" --unit="helloworld.service"
```

コマンドを直接実行するには、次の構文を使用します。この例では、スクリプト`/usr/local/bin/helloworld.sh`を直接呼び出します。

```
> sudo systemd-run --on-active="2hours" /usr/local/bin/helloworld.sh
```

コマンドがパラメータを取る場合は、スペースで区切って追加します。

```
> sudo systemd-run --on-active="2hours" /usr/local/bin/helloworld.sh --
language=pt_BR
```

過渡タイマは、単調タイマまたはリアルタイムタイマとすることができます。次のスイッチがサポートされており、単調タイマで説明されているように動作します。

- `--on-active`
- `--on-startup`
- `--on-unit-active`

- `--on-unit-inactive`
- `--on-calendar`

詳細については、**`man 1 systemd-run`**を参照してください。

5 カレンダエントリのテスト

`systemd`では、リアルタイムタイマ用のカレンダタイマエントリをテストおよび作成するためのツール、**`systemd-analyze calendar`**が提供されています。このツールでは、リアルタイムタイマの設定に必要な`OnCalendar`エントリと同じ引数を使用できます。

スペースで区切って複数の引数を連結できます。テストする期間が正しい場合は、タイマが次回トリガされる日時(ローカル時間とUTC)が出力に表示されます。また、`Normalized form`の文字列も表示されます。タイマファイルではこの文字列を使用することを推奨します。次に例を示します。

```
> systemd-analyze calendar "Tue,Sun *-*-* 01,03:00:00"
Normalized form: Tue,Sun *-*-* 01,03:00:00
Next elapse: Sun 2021-10-31 01:00:00 CEST
(in UTC): Sat 2021-10-30 23:00:00 UTC
From now: 3 days left

> systemd-analyze calendar "Mon..Fri *-*-* 10:00" "Sat,Sun *-*-* 22:00"
Original form: Mon..Fri *-*-* 10:00
Normalized form: Mon..Fri *-*-* 10:00:00
Next elapse: Thu 2021-10-28 10:00:00 CEST
(in UTC): Thu 2021-10-28 08:00:00 UTC
From now: 19h left

Original form: Sat,Sun *-*-* 22:00
Normalized form: Sat,Sun *-*-* 22:00:00
Next elapse: Sat 2021-10-30 22:00:00 CEST
(in UTC): Sat 2021-10-30 20:00:00 UTC
From now: 3 days left
```

繰り返しタイマでは、`–iterations N`スイッチを使用してトリガ時間を一覧表示し、想定どおりに動作するかどうかをテストします。引数`N`には、テストの繰り返し回数を指定します。次の例の文字列は、日曜日の00:00:00から8時間ごとにトリガすることを指定します。

```
> systemd-analyze calendar --iterations 5 "Sun *-*-* 0/08:00:00"
Original form: Sun *-*-* 0/08:00:00
Normalized form: Sun *-*-* 00/8:00:00
```

```
Next elapse: Sun 2021-10-31 00:00:00 CEST
(in UTC): Sat 2021-10-30 22:00:00 UTC
From now: 3 days left
Iter. #2: Sun 2021-10-31 08:00:00 CET
(in UTC): Sun 2021-10-31 07:00:00 UTC
From now: 3 days left
Iter. #3: Sun 2021-10-31 16:00:00 CET
(in UTC): Sun 2021-10-31 15:00:00 UTC
From now: 4 days left
Iter. #4: Sun 2021-11-07 00:00:00 CET
(in UTC): Sat 2021-11-06 23:00:00 UTC
From now: 1 week 3 days left
Iter. #5: Sun 2021-11-07 08:00:00 CET
(in UTC): Sun 2021-11-07 07:00:00 UTC
From now: 1 week 3 days left
```

6 タイマの失敗を通知する電子メールの取得

systemdには、cronのMAILTOに相当する機能が用意されていません。次の手順は、タイマの失敗を通知する電子メールを有効にする方法について説明しています。

この手順は次の各ステップで構成されます。

1. 電子メールを送信するスクリプトを作成します。
2. この電子メール送信スクリプトを実行するsystemdサービスファイルを作成します。
3. この電子メールサービスファイルをテストします。
4. タイマで制御するサービスから、OnFailureを使用して、作成した電子メールサービスファイルを呼び出します。

> 次の例では、mailxパッケージの**mailx**コマンドを使用しています。Postfix電子メールサーバがインストールされ、正しく設定されている必要があります。
>
> 1. スクリプト/usr/local/bin/send_systemd_emailを作成します。
>
>   a. このスクリプトには、$1 (電子メールアドレス)と$2 (失敗通知が受信されたサービスファイルの名前)の2つのパラメータが必要です。これらのパラメータはいずれも、メールスクリプトを実行するユニットファイルで指定します。
>
>    ```
>    #!/bin/sh
>    ```

```
systemctl status --full "$2" | mailx -S sendwait\
 -s "Service failure for $2" -r root@$HOSTNAME $1
```

b. スクリプトが実行可能であることを確認します。

```
> sudo chmod 755 /usr/local/bin/send_systemd_email
```

2. ファイル/etc/systemd/system/send_email_to_USER@.serviceを作成します。

```
[Unit]
Description=Send systemd status information by email for %i to USER

[Service]
Type=oneshot
ExecStart=/usr/local/bin/send_systemd_email EMAIL_ADDRESS %i
User=root
Group=systemd-journal
```

ファイル内のUSERとEMAIL_ADDRESSを、電子メールを受信するユーザのログイン名と電子メールアドレスに置き換えます。%iは、失敗したサービスの名前です(%nパラメータで電子メールサービスに渡されます)。

3. サービスファイルを確認して、報告された問題を修正します。

```
> systemd-analyze verify /etc/systemd/system/send_email_to_USER@.service
```

このコマンドから出力が返されなければ、ファイルは検証に適合しています。

4. 手順全体を確認するには、テスト用のdbusインスタンスを使用してサービスを開始します(現在実行している他のどのサービスも使用できます。あらゆるインストール環境でdbusサービスの動作が保証されているので、この例ではdbusを使用しています)。

```
> sudo systemctl start send_email_to_USER@dbus.service
```

サービスが正常に動作すれば、本文にdbusステータスメッセージが記述されて件名をService failure for dbusとした電子メールがEMAIL_ADDRESSに届きます(これはテストにすぎません。dbusサービスに問題は発生していないので、この電子メールを削除してもかまいません。どのようなアクションも不要です)。
テストの電子メールが正常に送信されていれば、それをサービスファイルに取り込んで作業を続けます。

5. サービスに電子メール通知を追加するには、失敗の通知対象とするサービスファイルのUnitセクションにOnFailureオプションを追加します。

```
[Unit]
Description="Hello World script"
```

```
OnFailure❶=send_email_to_USER❷@%n❸.service

[Service]
ExecStart=/usr/local/bin/helloworld.sh
```

❶ `OnFailure`オプションは、引数としてサービス名を取ります。

❷ サービスユニットファイル名をログイン名で置き換えます。

❸ サービスの名前を指定します(この例では`helloworld`)。電子メールサービスファイルでは、この名前が%iで表現されます。

これで、`systemd`サービスの失敗通知が正常に設定されました。

ヒント: 複数のユーザへの電子メール通知の送信

電子メールサービスファイルでは、受信者の電子メールアドレスがハードコーディングされています。別のユーザに通知電子メールを送信するには、電子メールサービスファイルをコピーして、ファイル名のユーザログインとコピー内の電子メールアドレスを置き換えます。

複数の受信者に失敗通知を同時に送信するには、次のように、サービスファイルにそれぞれのサービスファイル名をスペースで区切って追加します。

```
OnFailure=send_email_to_tux@%n.service send_email_to_wilber@%n.service
```

7 通常のユーザとしてのタイマの使用

通常のユーザでも`systemd`タイマを使用できます。バックアップ、イメージの処理、クラウドへのデータの移動など、繰り返し実行するタスクの自動化で効果的です。

システム規模タイマの場合と同じ手順とタスクが有効です。ただし、次のような違いがあります。

- タイマとサービスファイルは`~/.config/systemd/user/`に配置する必要があります。
- すべての**`systemctl`**コマンドと**`journalctl`**コマンドは、`--user`スイッチを使用して実行する必要があります。**`systemd-analyze`**では、このオプションは不要です。
  通常のユーザは、次の例のようにユニットファイルへのパスを指定する必要があります。これを指定しない場合、同じ名前のシステム規模タイマが存在すると、それが代わりに実行されるか列挙されます。

```
> systemctl --user start ~/.config/systemd/user/helloworld.timer
> systemctl --user enable ~/.config/systemd/user/helloworld.timer
> systemctl --user list-timers
> journalctl --user -u helloworld.*
> systemd-analyze verify ~/.config/systemd/user/helloworld.timer
```

重要: ユーザタイマはアクティブなセッションでのみ実行可能

通常のユーザとして開始した他のsystemdサービスと同様に、ユーザタイマはユーザがログインしているときにのみ実行されます。代わりに、ブート時にユーザタイマを開始し、ログアウト後も実行し続けるには、影響を受ける各ユーザに対して「リンガ」を有効にします。

```
sudo loginctl enable-linger USER
```

詳細については、**man 1 loginctl**を参照してください。

重要: 環境変数は非継承

systemdのユーザインスタンスでは、~/.profileや~/.bashrcなどのスクリプトで設定した環境変数が継承されません。systemdの環境を確認するには**systemctl --user show-environment**を実行します。

systemdの環境にない変数をインポートするには、~/.bashrcの末尾で次のコマンドを指定します。

```
systemctl --user import-environment VARIABLE1 VARIABLE2
```

8 cronからsystemdタイマへの移行

すべてのcronジョブをsystemdのタイマへ移行できます。手順と例はこちらをご覧ください。

1. スクリプトを実行するサービスファイルを作成します。詳細については例1「サービスファイル」を参照してください。
2. サービスファイルを実行するタイマファイルを作成します。一般的な手順については、例2「タイマファイル」を参照してください。

a. カレンダのエントリを変換します。時間はcronとsystemdでは指定方法が異なります。次のパターンを変換テンプレートとして使用します。

```
Cron:               Minute Hour Day Month DayOfWeek
systemd: OnCalendar=DayOfWeek Year-Month-Day Hour:Minute:Second
```

変換したカレンダエントリをテストするには5項 「カレンダエントリのテスト」の手順に従います。

b. 次のようにcronのニックネーム(@NICK)を変換します。

```
Cron     : systemd timer
-------- : ---------------------------
@reboot  : OnBootSec=1s
@yearly  : OnCalendar=*-01-01 00:00:00
@annually: OnCalendar=*-01-01 00:00:00
@monthly : OnCalendar=*-*-01 00:00:00
@weekly  : OnCalendar=Sun *-*-* 00:00:00
@daily   : OnCalendar=*-*-* 00:00:00
@hourly  : OnCalendar=*-*-* *:00:00
```

c. 変数の割り当てを変換します。systemdでの変数の割り当ては[Service]セクションに記述する必要があります。MAILTOは、この方法では変換できません。その変換方法については次のステップを参照してください。

```
cron: VARIABLE=VALUE
systemd: Environment="VARIABLE=VALUE"
```

d. 6項 「タイマの失敗を通知する電子メールの取得」の手順に従って、cronのMAILTO機能を置き換える電子メール通知を設定します。

例 6: CRONからsystemdのタイマへの移行

以下は、ブートの5分後と、毎週月曜日～金曜日の10時にスクリプトhelloworld.shを呼び出すcrontabエントリです。

```
@reboot sleep 300 && /usr/local/bin/helloworld.sh
0 10 * * * 1-5 /usr/local/bin/helloworld.sh
```

このスクリプトを呼び出すsystemdのサービスファイル(helloworld.service)は次のようになります。

```
[Unit]
Description="Hello World script"
[Service]
ExecStart=/usr/local/bin/helloworld.sh
```

タイマファイル(helloworld.timer)は次のようになります。

```
[Unit]
Description="Run helloworld.service 5min after boot and at 10am every Mon-Fri"
[Timer]
OnBootSec=5min
OnCalendar=Mon..Fri *-*-* 10:00:*
Unit=helloworld.service
[Install]
WantedBy=multi-user.target
```

9 トラブルシューティングとFAQ

失敗したsystemdタイマをデバッグおよびトラブルシューティングする方法を説明します。systemdのタイマに関するFAQ (よくある質問と答え)の回答を参照してください。

9.1 エラーの回避

systemdのタイマでエラーを回避するには次のベストプラクティスに従います。

- ExecStartを使用してサービスで指定した実行可能ファイルを正しく実行できることを確認します。
- **systemd-analyze verify** FILEを実行して、サービスとタイマファイルの構文を確認します。
- **systemd-analyze calendar** CALENDER_ENTRYを実行して、カレンダのエントリの実行時間を確認します。

9.2 イベントがトリガされない

systemdでは、重大ではないエラーがあるタイマを有効にすると、それらのエラーが暗黙的に無視されます。次に例を示します。

例 7: 致命的ではないエラーがあるsystemdタイマファイルの抜粋

```
[Timer]
```

```
OnBootSec=5min
OnClendar=Mon..Fri 10:00
Unit=helloworld.service
```

3行目に、`OnCalendar`ではなく`OnClendar`を使用している構文エラーがあります。`[Timer]`セクションに2つ目のタイマエントリ(OnBoot)が含まれるため、このエラーは重大ではなく、無視されて何も表示されません。その結果、月曜日から金曜日までのトリガが実行されません。このエラーを検出するには**`systemd-analyze verify`**コマンドを使用する以外にありません。

```
#  systemd-analyze verify /etc/systemd/system/helloworld.timer
/etc/systemd/system/helloworld.timer:7: Unknown key name 'OnClendar' in section
 'Timer', ignoring.
```

9.3 システムジャーナルのエラー確認

すべての`systemd`サービスと同様に、タイマによってトリガされたイベントやアクションは、システムジャーナルに記録されます。トリガが想定どおりに動作しない場合は、**`journalctl`**を使用してログメッセージを確認します。ジャーナルをフィルタして関連する情報を表示するには、`-u`スイッチを使用して`systemd`タイマとサービスファイルを指定します。このオプションを使用して、タイマおよび対応するサービスファイルのログエントリを表示します。

```
sudo journalctl -u  helloworld.timer -u helloworld.service
```

短縮形で十分な場合は次のとおりです。

```
sudo journalctl -u  helloworld.*
```

`journalctl`は、さまざまなオプションとフィルタをサポートするツールです。詳細については、**`man 1 journalctl`**を参照してください。次のオプションはタイマのトラブルシューティングで効果的です。

- `-b`: 現在のブートのエントリのみを表示します。
- `-S today`: 今日からのエントリのみを表示します。
- `-x`: ログエントリの隣にヘルプテキストを表示します。
- `-f`: 最新のエントリから開始し、新しいエントリが追加されるたびにログを継続的に出力します。短い間隔で発生するトリガの確認で有用です。 `Ctrl` – `C` を押すと終了します。

9.4 systemdのタイマ: 実行されなかったトリガの再実行

systemdのタイマに想定したトリガ時刻にタイマが非アクティブであったかシステムがオフ状態であった場合、タイマを再度有効にすると、イベントに対して実行されなかったトリガを必要に応じて実行できます。この機能を有効にするには、次のように[Timer]セクションに設定オプションPersistent=trueを追加します。

```
[Timer]
OnCalendar=Mon..Fri 10:00
Persistent=true
Unit=helloworld.service
```

9.5 cronからsystemdのタイマへ移行する方法

すべてのcronジョブをsystemdのタイマへ移行できます。ここでは、cronジョブを移行するための一般的な手順を示します。

1. スクリプトを実行するサービスファイルを作成します。詳細については例1「サービスファイル」を参照してください。
2. サービスファイルを実行するタイマファイルを作成します。一般的な手順については、例2「タイマファイル」を参照してください。

    a. カレンダのエントリを変換します。時間はcronとsystemdでは指定方法が異なります。次のパターンを変換テンプレートとして使用します。

    ```
    Cron:               Minute Hour Day Month DayOfWeek
    systemd: OnCalendar=DayOfWeek Year-Month-Day Hour:Minute:Second
    ```

    変換したカレンダエントリをテストするには5項「カレンダエントリのテスト」の手順に従います。

    b. 次のようにcronのニックネーム(@NICK)を変換します。

    ```
    Cron     : systemd timer
    -------- : ---------------------------
    @reboot  : OnBootSec=1s
    @yearly  : OnCalendar=*-01-01 00:00:00
    @annually: OnCalendar=*-01-01 00:00:00
    @monthly : OnCalendar=*-*-01 00:00:00
    @weekly  : OnCalendar=Sun *-*-* 00:00:00
    @daily   : OnCalendar=*-*-* 00:00:00
    @hourly  : OnCalendar=*-*-* *:00:00
    ```

c. 変数の割り当てを変換します。systemdでの変数の割り当ては[Service]セクションに記述する必要があります。MAILTOは、この方法では変換できません。その変換方法については次のステップを参照してください。

```
cron: VARIABLE=VALUE
systemd: Environment="VARIABLE=VALUE"
```

d. 6項 「タイマの失敗を通知する電子メールの取得」の手順に従って、cronのMAILTO機能を置き換える電子メール通知を設定します。

例 8: CRONからsystemdのタイマへの移行

以下は、ブートの5分後と、毎週月曜日～金曜日の10時にスクリプトhelloworld.shを呼び出すcrontabエントリです。

```
@reboot sleep 300 && /usr/local/bin/helloworld.sh
0 10 * * * 1-5 /usr/local/bin/helloworld.sh
```

このスクリプトを呼び出すsystemdのサービスファイル(helloworld.service)は次のようになります。

```
[Unit]
Description="Hello World script"
[Service]
ExecStart=/usr/local/bin/helloworld.sh
```

タイマファイル(helloworld.timer)は次のようになります。

```
[Unit]
Description="Run helloworld.service 5min after boot and at 10am every Mon-Fri"
[Timer]
OnBootSec=5min
OnCalendar=Mon..Fri *-*-* 10:00:*
Unit=helloworld.service
[Install]
WantedBy=multi-user.target
```

10 詳細情報

- 高度な設定オプション(遅延やクロックの処理やタイムゾーンの変更など)を含むsystemdタイマの完全な参照については、**man 5 systemd.timer**を参照してください。
- Basic systemd concepts (https://documentation.suse.com/smart/systems-management/html/concept-systemd/concept-systemd.html)

- Starting and stopping `systemd` services (https://documentation.suse.com/smart/systems-management/html/reference-systemctl-start-stop-services/reference-systemctl-start-stop-services.html)
- Enabling and disabling `systemd` services (https://documentation.suse.com/smart/systems-management/html/reference-systemctl-enable-disable-services/reference-systemctl-enable-disable-services.html)
- Debugging failed `systemd` services (https://documentation.suse.com/smart/systems-management/html/task-debug-failed-systemd-services/index.html)
- Sending termination signals to `systemd` services (https://documentation.suse.com/smart/systems-management/html/task-send-termination-signals-systemd/task-send-termination-signals-systemd.html)

11 法的事項

A GNUフリー文書利用許諾契約書(GFDL)

0. 序文

この利用許諾契約書の目的は、マニュアル、テキストブック、またはその他の機能的で有用な文書を、自由という意味で「フリー」にすることです。つまり、そのような文書を、変更の有無や商用非商用に関わらず、コピーまたは再配布する実効的な自由をすべての人々に保証することです。第二に、本利用許諾契約書は、作者または発行者が他者によって行われた変更について責任を負わないとともに、その著作物の功績が確保されるように意図されています。

本利用許諾契約書は、「コピーレフト」(著作物を自由に複製および改変できるようにすること)の一種であり、文書の派生著作物は、それ自体が同じ意味においてフリーでなければなりません。フリーソフトウェア向けに考慮されたコピーレフト利用許諾であるGNU一般公衆利用許諾契約書(GPL)を補足するものです。

弊社は、この利用許諾契約書をフリーソフトウェアのマニュアルに使用するために設計しました。それは、フリーソフトウェアにはフリーマニュアルが必要であるためです。つまり、フリープログラムには、そのソフトウェアと同じ自由を提供するマニュアルが付属しなければなりません。ただし、本利用許諾契約書は、ソフトウェアマニュアルに制限されるものではありません。主題であるか否か、または印刷された本として発行されるか否かに関わらず、任意のテキスト著作物に使用することができます。本利用許諾契約書は、その目的が指示または参照に置かれている著作物に主に使用することを推奨します。

1. 適用範囲と定義

本利用許諾契約書は、この利用許諾の条項に従って頒布できることを定めた著作権者の通告が記載されている任意のメディアにおけるマニュアルまたは他の著作物に適用されます。そのような通告は、その著作物をここに記載されている条件に従って使用するための世界的な無償の利用許諾を無期限で付与します。次に示す「文書」は、そのような任意のマニュアルまたは著作物を指します。その公衆ユーザはいずれも被許諾者であり、「利用者」と呼ばれます。利用者は、著作権法に従った許可が必要になるような方法で著作物を複製、変更または頒布する場合に、利用許諾を受け入れます。

文書の「変更された版」とは、そのまま複製されるか、変更または別の言語に翻訳された(またはその両方)文書あるいはその一部を含んだ著作物のことです。

「二次セクション」は、文書の発行者または作者と文書の全体的な主題(または関連事項)との関係のみを示す文書の名前付き付録または前付け部分です。総体的な主題に直接関わる内容は含まれていません。(したがって、文書が部分的に数学のテキストブックになっている場合、二次セクションでは数学について説明されない場合があります)。関係には、主題または関連事項との歴史的なつながり、あるいはそれらに関する法的、商的、哲学的、倫理的、政治的位置付けが含まれる場合があります。

「不変セクション」は、文書が本利用許諾契約書の条件の下でリリースされる旨を述べている通告において、そのタイトルが不変セクションのものとして指定されている、ある特定の二次セクションです。セクションが、すでに説明した二次セクションの定義に一致しない場合は、不変として指定することはできません。文書には、不変セクションが含まれない場合があります。文書で不変セクションを特定しない場合、不変セクションは含まれません。

「カバーテキスト」とは、文書が本利用許諾契約書の条件の下でリリースされる旨を述べている通告において、表カバーテキストまたは裏カバーテキストとして列挙されている、ある一定の短い文章のことです。表カバーテキストは、最大で5語、裏カバーテキストは、最大で25語によって構成できます。

文書の「透過的な複製」とは、その仕様が一般の利用者にとって入手可能で、一般的なテキストエディタまたは一般的な描画プログラム(画素で構成される画像用)、あるいは広く使用されている図面エディタ(図面用)で文書を直接改訂するのに適した形式で表される機械可読の複製のことです。テキストフォーマッタへの入力またはテキストフォーマッタへの入力に適したさまざまな形式への変換に適していることも前提になります。読者による以後の変更を阻止または妨げるようにマークアップまたはマークアップのない状態が調整されている、他の点では透過的なファイル形式で行われた複製は、透過的な複製ではありません。イメージ形式は、相当量のテキストに使用されている場合、透過的ではありません。「透過的」ではない複製は、「不透明」と呼ばれます。

透過的な複製に適した形式として、マークアップのないプレーンなASCII、Texinfo入力形式、LaTeX入力形式、一般に取得可能なDTDを使用するSGMLまたはXML、標準に準拠したHTML、人為的変更用のPostScriptまたはPDFがあります。透過的なイメージ形式には、PNG、XCF、JPGが含まれます。不透明な形式には、独自のワードプロセッサのみで読み取りおよび編集を行える独自の形式、DTDまたは処理(またはその両方)ツールを一般に取得できないSGMLまたはXML、機械生成HTML、出力のみを目的として一部のワードプロセッサによって作成されるPostScriptまたはPDFが含まれます。

「タイトルページ」とは、印刷された本の場合、タイトルページ自体、および本利用許諾契約書でタイトルページに表示することが要求されるマテリアルを読みやすいように保持するために必要な以降のページのことを指します。そのようなタイトルページがない形式の著作物の場合、「タイトルページ」は、本文の開始部分に先行する、著作物のタイトルを最も顕著に表している部分の近くにあるテキストのことを指します。

「XYZという表題の付いた」セクションとは、そのタイトルが正確にXYZになっているか、またはXYZを別の言語に翻訳しているテキストに続いてカッコ付きのXYZが含まれている文書の名前付きサブユニットのことです。(ここで、XYZは、次に示すように、「謝辞」、「献辞」、「推薦」、「履歴」などの特定のセクション名を表します)。文書を変更するときに、そのようなセクションの「タイトルを保存する」とは、この定義に従って「XYZという表題の付いた」セクションが残されることを表します。

文書では、本利用許諾契約書が文書に適用される旨を述べている通告の付近に保証の放棄を含めることができます。保証の放棄条項は、本利用許諾契約書内の参照によって、保証の放棄に関してのみ組み込まれると見なされます。つまり、これらの保証の放棄条項がもつ可能性のある他のいかなる含意も無効であり、本利用許諾契約書の意味にまったく影響を与えません。

2. そのままの複製

利用者は、商用か否かを問わず、任意のメディアにおいて文書を複製または頒布することができます。その際に、本利用許諾契約書、著作権表示、および本利用許諾契約書が文書に適用される旨を述べる利用許諾通告をすべての複製で再生し、本利用許諾契約書の条件に他のいかなる条件も追加しないことが前提条件になります。利用者は、技術的手段によって、作成または頒布する複製の読み込みまたはさらなる複製を妨げたり、制御したりすることはできません。ただし、複製と引き換えに対価を受け取ることができます。十分に大量の複製を頒布する場合は、セクション3の条件に従う必要もあります。

すでに述べた同じ条件に従って複製を貸与したり、複製を公開したりすることもできます。

3. 大量の複製

発行する文書の印刷した複製(または、通常、印刷したカバーをもつメディアに含まれた複製)が 100部を超え、文書の利用許諾通告でカバーテキストを必要とする場合は、すべてのカバーテキスト(表カバーの表カバーテキスト、裏カバーの裏カバーテキスト)を明瞭かつ読みやすく記載したカバーに文書の複製を同封する必要があります。また、両方のカバーでは、これらの複製の発行者として、利用者を読みやすい状態で明確に識別しなければなりません。表カバーには、フルタイトルを記述し、タイトルのすべての語が同等に目立つようにする必要があります。カバーには他のマテリアルを追加することもできます。カバーに限って変更を行った場合の複製は、文書のタイトルが保持されていて、これらの条件を満たしている限り、他の点に関してそのままの複製と見なすことができます。

いずれかのカバーで、必要なテキストが多すぎて、読みやすい状態に収まらない場合は、列挙されている最初の部分(問題なく収まる分)を実際のカバーに記載し、残りの部分を隣接ページに入れます。

文書の不透明な複製を100部以上公開または頒布する場合は、それぞれの不透明な複製とともに機械可読の透過的な複製を含めるか、それぞれの不透明な複製内あるいはその複製とともに、ネットワークの一般利用者が標準的な一般ネットワークプロトコルを使用して、追加マテリアルのない文書の完全な透過的複製をダウンロードするときにアクセスできるコンピュータネットワークの場所を明記する必要があります。後者のオプションを使用する場合は、不透

明な複製の大量頒布を開始するときに十分慎重な手順を取り、この透過的な複製が、その版の不透明な複製を最後に一般頒布した後(直接またはエージェントや小売業者を通じて)少なくとも1年間、指定した場所で継続的にアクセス可能となるように配慮する必要があります。

大量の複製を再頒布する時点よりもかなり前に、文書の作者に連絡して、文書の更新版を提供する機会を与えることが要求されますが、必須ではありません。

4. 変更

文書の変更された版を、すでに述べた第2項および第3項の条件に従って複製および頒布することができます。その際は、本利用許諾契約書に確実に従って、変更された版をリリースし、変更された版が文書の役割を担うようにして、その複製を所要する任意の利用者に変更された版の頒布および変更の利用許諾を与えることが前提になります。また、変更された版で次のことを行う必要があります。

A. タイトルページ(カバーがある場合はカバー上も含める)で、文書、および以前の版の文書(以前の版がある場合は、その旨、文書の履歴セクションに列挙する)と識別されるタイトルを使用します。前の版と同じタイトルは、その版の元の発行者が許可を与えた場合に、使用することができます。

B. タイトルページ上に、この要件から解放されない限り、変更された版において変更の著者としての責任を担う1人以上の人またはエンティティとともに、文書の筆頭著者を少なくとも5人、作者として列挙します(5人に満たない場合は、その筆頭著者のすべて)。

C. タイトルページ上に、変更された版の発行者の名前を、発行者として記載します。

D. 文書のすべての著作権表示を保持します。

E. 変更に関する適切な著作権表示を、他の著作権表示の隣に追加します。

F. 著作権表示の直後に、本利用許諾契約書の条項に従って変更された版を利用するための許可を一般利用者に与える利用許諾通告を、次の補遺に示す形式で含めます。

G. その利用許諾通告に、不変セクションの全リスト、および文書の利用許諾通告で指定されている必須カバーテキストを保持します。

H. 本利用許諾契約書の変更されていない複製を含めます。

I. 「履歴」という表題のセクションを保持して、そのタイトルを保持し、タイトルページに記載されているとおりに、変更された版のタイトル、年度、新しい作者、発行者を少なくとも示す項目を追加します。文書に履歴というセクションがない場合は、そのタイトルページに記載されているとおりに文書のタイトル、年度、作者、発行者を示すセクションを作成し、前の文章に記載されているとおりに変更された版を示す項目を追加します。

J. 文書の透過的な複製に一般利用者がアクセスできるように文書で指定されている場合は、そのネットワークの場所、およびその文書の基盤となった前の版に対応して文書で指定されているネットワークの場所を保持します。これらは、「履歴」セクションに配置することができます。文書自体よりも4年以上前に発行された著作物の場合、または参照されているその版の元の発行者が許可を与えている場合は、そのネットワークの場所を省略することができます。

K. 「謝辞」または「献辞」という表題のセクションの場合は、そのセクションのタイトルを保持し、セクション内に、それぞれの貢献者謝辞またはその中の献辞(またはその両方)のすべての内容と意味合いを保持します。

L. 文書のすべての不変セクションを保持し、そのテキストおよびタイトルを未変更のままにします。セクション番号またはそれと同等の要素は、セクションタイトルの一部と見なされません。

M. 「推薦」という表題の任意のセクションを削除します。そのようなセクションは、変更された版に含めることはできません。

N. 既存のセクションのタイトルを変更して、「推薦」という表題にしたり、タイトルが不変セクションと矛盾したりしないようにします。

O. 保証の放棄を保持します。

変更された版に、二次セクションと見なされ、文書から複製されたマテリアルを含まない新しい前付けセクションまたは付録が含まれる場合は、これらの一部またはすべてを任意に「不変」として指定することができます。これを行うには、変更された版の利用許諾表示内で列挙されている不変セクションにそのタイトルを追加します。これらのタイトルは、他のすべてのセクションタイトルと異なっている必要があります。

「推薦」という表題のセクションを追加することができますが、その際は、変更された版のさまざまな当事者による推薦以外の要素が含まれていないことが前提になります。たとえば、校正者によるコメント、または文が標準的な信頼できる定義として組織によって承認されていることを示すという宣言文などが相当します。

表カバーテキストとしての最大5語の短い文、および裏カバーテキストとしての最大25語の短い文を、変更された版のカバーテキストのリストの終わりに追加できます。表カバーテキストの短い1文および裏カバーテキストの短い1文のみを、1つのエンティティが追加できます(またはエンティティによって行われた調整を通じて)。文書に、利用者または利用者が関わる同じエンティティによって行われた調整を通じて前に追加された同じカバーのカバーテキストがすでに含まれている場合は、別のものを追加することはできませんが、古いものを置き換えることができます。ただし、その古いものを追加した前の発行者から明示的な許可を得る必要があります。

文書の作者および発行者は、本利用許諾契約書により、その名前を得るために使用したり、変更された版の推薦を主張または暗示したりする許可を与えるものではありません。

5. 文書の結合

文書は、すでに述べた変更された版に関するセクション4の条件に従って、本利用許諾契約書の下でリリースされた他の文書と結合することができます。その際は、その組み合わせの中に、元の全文書のすべての不変セクションを未変更のまま含めて、そのすべてを結合された著作物の不変セクションとしてその利用許諾表示に列挙し、そのすべての保証の放棄を保持することが前提となります。

結合された著作物には、本利用許諾契約書の複製を1つのみ含める必要があります。複数の同一の不変セクションは、単一の複製で置き換えることができます。同じ名前だが内容の異なる複数の不変セクションがある場合は、そのような各セクションのタイトルを固有なものにします。その際は、その終わりに、カッコ付きで、そのセクションの元の作者または発行者の名前(既知の場合)、あるいは固有の番号を追加します。不変セクションのリスト内のセクションタイトルには、結合された著作物の利用許諾表示の場合と同じ調整を加えます。

組み合わせでは、さまざまな元の文書の「履歴」という表題のセクションを結合して、1つの「履歴」というセクションを構築する必要があります。同じように、「謝辞」という表題のセクション、および「献辞」という表題のセクションも結合します。「推薦」という表題のすべてのセクションを削除する必要があります。

6. 文書のコレクション

文書および本利用許諾契約書の下でリリースされた他の文書から成るコレクションを作成して、さまざまな文書に含まれる本利用許諾契約書の個々の複製を、コレクションに含まれる単一の複製で置き換えることができますが、他のすべての点での各文書のそのままの複製に関する本利用許諾契約書の規則に従うことが前提になります。

そのようなコレクションから単一の文書を抽出して、その文書を本利用許諾契約書に従って個々に頒布することができますが、その際は、本利用許諾契約書の複製を抽出した文書に挿入して、その文書のそのままの複製に関するその他のすべての点で、本利用許諾契約書に従う必要があります。

7. 独立した著作物の集積

文書またはその派生物を他の個別および独立した文書または著作物とともに、ストレージまたは頒布メディア内またはそのボリューム上に蓄積することを「集積」と呼びます。その場合は、個々の著作物の許可を超えてその蓄積の利用者の法的権限を制限することに、蓄積による著作権を使用しないことが前提になります。文書が集積に含まれる場合、本利用許諾契約書は、それ自体が文書の派生著作物ではない集積内の他の著作物に適用されません。

セクション3のカバーテキスト要件が文書のこれらの複製に適用可能であり、文書が集積全体の半分に満たない場合は、文書のカバーテキストを、集積内の文書のカバー、または文書が電子形式の場合は、電子的な同等のカバーに配置することができます。それ以外の場合は、集積全体の印刷されたカバー上に表示する必要があります。

8. 翻訳

翻訳は一種の変更と見なされるため、セクション4の条件に従って文書の翻訳を配布することができます。不変セクションを翻訳で置き換えるには、著作権者からの特別な許可が必要ですが、これらの不変セクションの元のバージョンのほかに、一部またはすべての不変セクションの翻訳を含めることができます。本利用許諾契約書、文書内のすべての利用許諾表示、および保証の放棄の翻訳を含めることができますが、その際は、本利用許諾契約書の元の英語版、およびそれらの利用許諾表示と保証の放棄の元の版も含めることが前提になります。本利用許諾契約書、利用許諾表示、または保証の放棄の翻訳と元の版との間に不一致がある場合は、元の版が優先されます。

文書内のセクションに、「謝辞」、「献辞」、または「履歴」という表題が付いている場合は、そのタイトルを保持する(セクション1)ための要件(セクション4)により、通常は実際のタイトルを変更する必要があります。

9. 終了

利用者は、本利用許諾契約書に明示的に記載されている形態を除き、文書を複製、改変、二次利用許諾、および頒布してはなりません。別の方法で文書を複製、改変、二次利用許諾、または頒布しようとするのは無効であり、本利用許諾契約書の下で利用者の権利は自動的に消滅します。ただし、本使用許諾の下で利用者から複製物または権利を受領した関係者は、条項を遵守している限り、権利が消滅することはありません。

10. 本利用許諾契約書の将来の改訂

フリーソフトウェア財団は、GNUフリー文書利用許諾契約書(GFDL)の新版または改訂版を随時公表することがあります。そのような新版は、性格的には現行版と似たものになりますが、新たな問題や懸案事項に対応するために細部が異なる可能性があります。https://www.gnu.org/copyleft/を参照してください。

本利用許諾契約書の各版には、区別するための版番号が設定されます。文書に、それに適用される本利用許諾契約書の特定の版番号と「後継版」が指定されている場合、利用者は、選択によって、その指定された版の条項またはフリーソフトウェア財団から公開される後継版の条項(ドラフトではない)に従うことになります。文書に、本利用許諾契約書の版番号が指定されていない場合、利用者は、フリーソフトウェア財団からこれまでに公開された任意の版(ドラフトではない)を選択することができます。

補遺: 本利用許諾書をご使用の文書に使用する方法

```
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 1.2
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”.
```

不変セクション、表カバーテキスト、および裏カバーテキストがある場合は、「with...Texts」の行を次のように置き換えます。

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

カバーテキストのない不変セクションが含まれている場合、またはこの3つの他の組み合わせの場合は、その2つの代替要素をマージして状況に合わせます。

文書にプログラムコードの重要な例が含まれている場合は、GNU一般公衆利用許諾契約書(GPL)などの選択したフリーソフトウェアの利用許諾に従って、これらの例を平行してリリースし、フリーソフトウェアでのその利用を許可することを推奨します。