Ubuntu サーバーの特徴とさまざまな機能

本日の発表資料：
https://wiki.ubuntu.com/MitsuyaShibata/Slides

- 発表資料は OSC のページからもアクセスできる予定です
- 質問・ツッコミは思いついたタイミングでお願いします
- つまり「ご静聴」しないほうがうれしいです
- 発表資料は CC BY-SA 4.0[1]で提供します
- 社内での布教などご自由にお使いください

[1] https://creativecommons.org/licenses/by-sa/4.0/deed.ja

本日の発表内容

「**適当な** Linux サーバー用意しといて」
「ディストリビューションは何でもいいですか？」
「**取り敢えず** Ubuntu で」

……などと言われてしまった**不幸な子羊**にむけて
Ubuntu サーバーの最新情報や基本的な特徴をお伝えします。

本日の結論

本日の結論

Ubuntu も
インストールすれば
ただの Linux

要するに：「サーバーでも Ubuntu を使ってみよう！」

- ちょっとだけ、**お試し**でいいから始めてみようよ
- インストールするだけなら**無料**なんだしさ
- **みんな**使っているよ
- **安全**にも気を使っているから、**安心**して
- **社会勉強**みたいなものだと思って、どう？
- サポート代も（全部自己解決すれば）**実質無料**だよ

要するに：「サーバーでも Ubuntu を使ってみよう！」

- ちょっとだけ、**お試し**でいいから始めてみようよ
- インストールするだけなら**無料**なんだしさ
- **みんな**使っているよ
- **安全**にも気を使っているから、**安心**して
- **社会勉強**みたいなものだと思って、どう？
- サポート代も（全部自己解決すれば）**実質無料**だよ

要するに：「サーバーでも Ubuntu を使ってみよう！」

- ちょっとだけ、**お試し**でいいから始めてみようよ
- インストールするだけなら**無料**なんだしさ
- **みんな**使っているよ
- **安全**にも気を使っているから、**安心**して
- **社会勉強**みたいなものだと思って、どう？
- サポート代も（全部自己解決すれば）**実質無料**だよ

要するに：「サーバーでも Ubuntu を使ってみよう！」

- ちょっとだけ、**お試し**でいいから始めてみようよ
- インストールするだけなら**無料**なんだしさ
- **みんな**使っているよ
- **安全**にも気を使っているから、**安心**して
- **社会勉強**みたいなものだと思って、どう？
- サポート代も（全部自己解決すれば）**実質無料**だよ

要するに：「サーバーでも Ubuntu を使ってみよう！」

- ちょっとだけ、**お試し**でいいから始めてみようよ
- インストールするだけなら**無料**なんだしさ
- **みんな**使っているよ
- **安全**にも気を使っているから、**安心**して
- **社会勉強**みたいなものだと思って、どう？
- サポート代も（全部自己解決すれば）**実質無料**だよ

要するに：「サーバーでも Ubuntu を使ってみよう！」

- ちょっとだけ、**お試し**でいいから始めてみようよ
- インストールするだけなら**無料**なんだしさ
- **みんな**使っているよ
- **安全**にも気を使っているから、**安心**して
- **社会勉強**みたいなものだと思って、どう？
- サポート代も（全部自己解決すれば）**実質無料**だよ

Ubuntu の紹介

Ubuntu とは

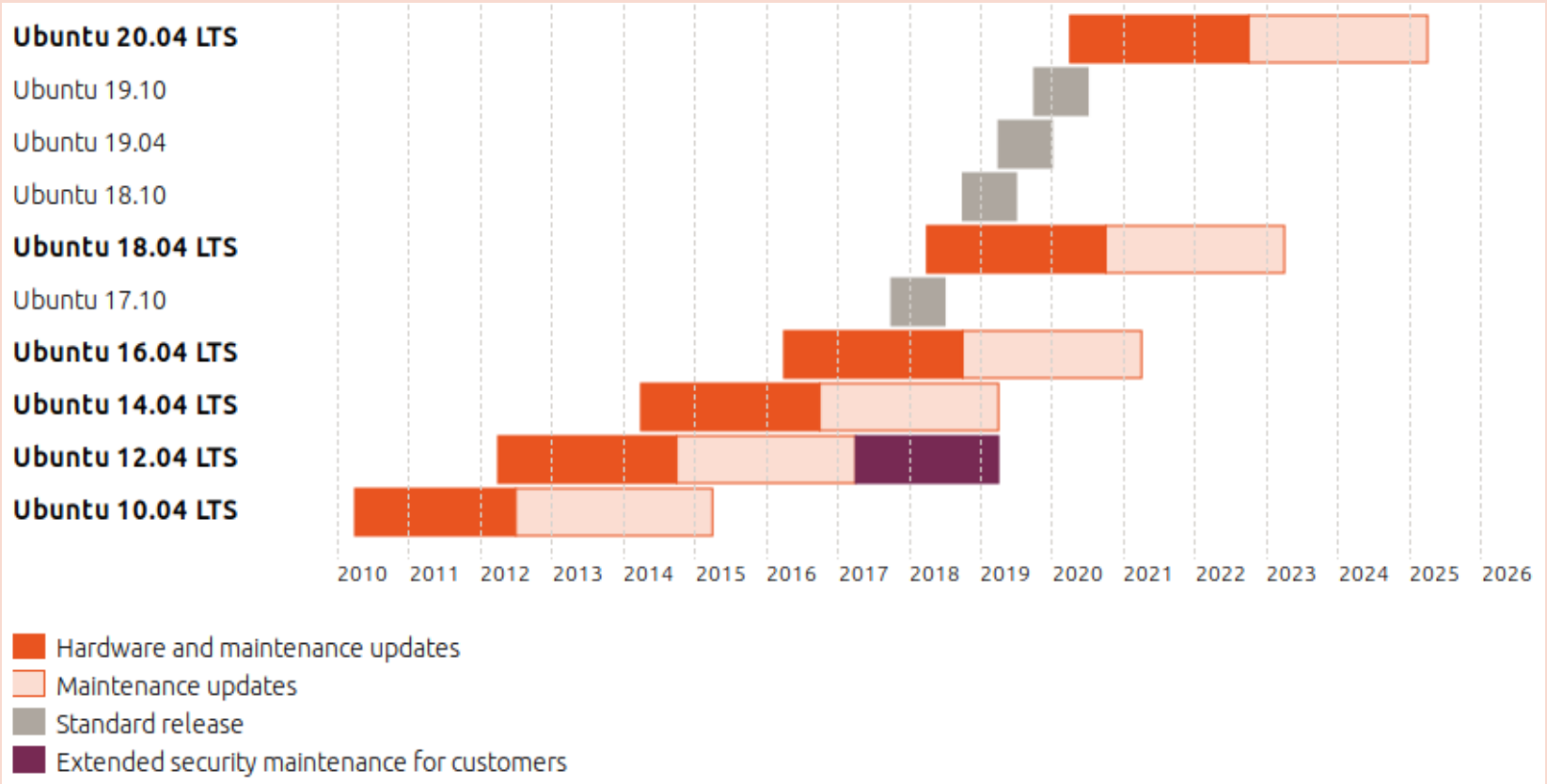
- **Debian ベース**の Linux ディストリビューション
- ターゲット：デスクトップ／サーバー／クラウド／ IoT
- 開発の主体は Ubuntu コミュニティ
- Canonical はそれを支援しているという形
- 「商用版」は存在しない
- Canonical が商用サポートサービスを提供している

リリースはタイムベース

- **4 月**と **10 月**、半年に 1 度のタイムベースリリース
- バージョンは「西暦下二桁. リリース月」
- **18.04** は **2018 年 4 月**で **18.10** は **2018 年 10 月**にリリース
- 2 年に 1 度の 4 月に**長期サポート版（LTS）**をリリース

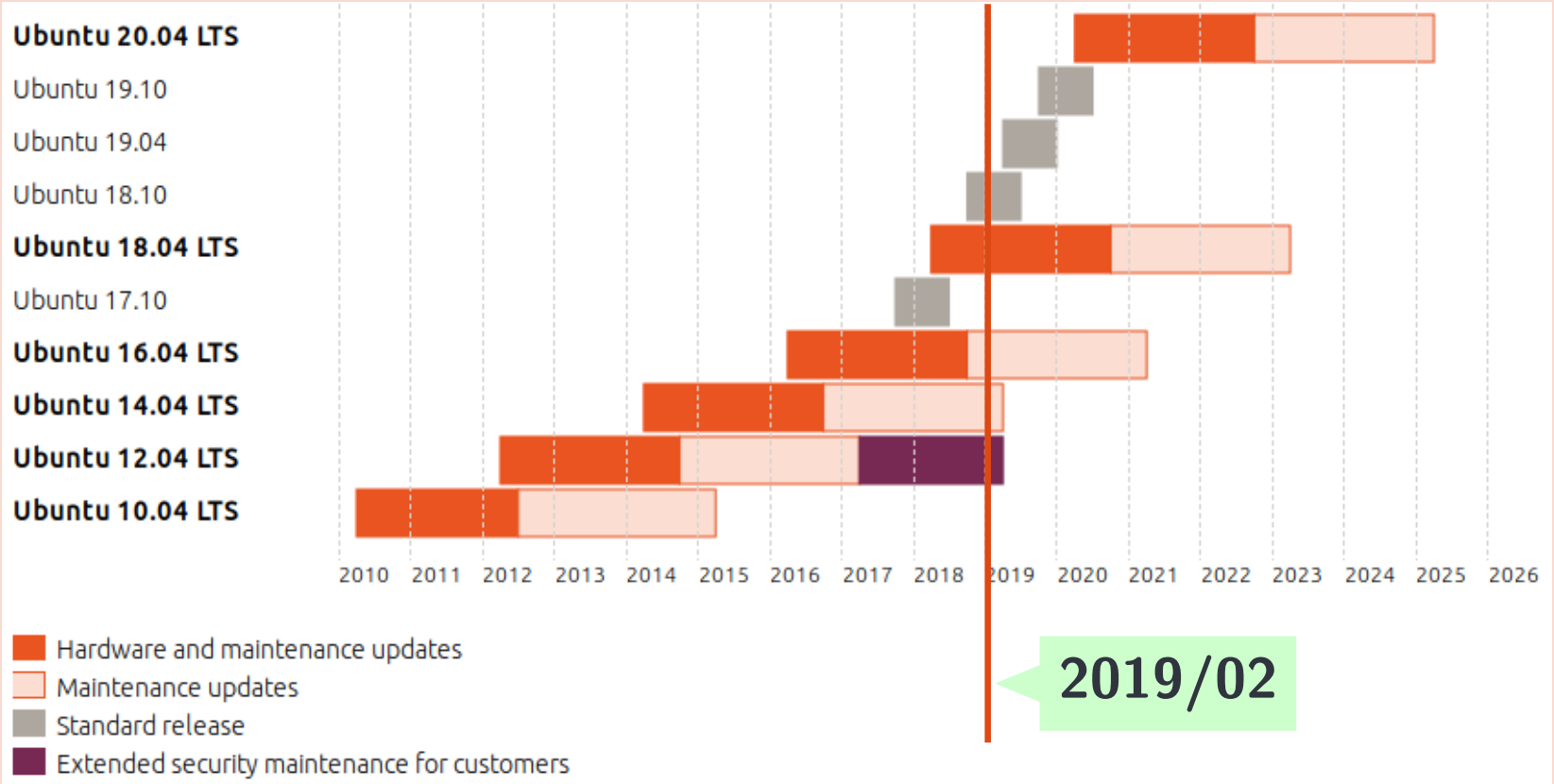
間もなく 14.04 のサポートが終了します

リリースはタイムベース



https://www.ubuntu.com/about/release-cycle

リリースはタイムベース



https://www.ubuntu.com/about/release-cycle

長期サポート版（LTS）

- 通常リリースは**9ヶ月**のサポート期間
- 6ヶ月ごとにリリースなので常にアップグレードが必要
- LTSは**5年**のサポート期間
- 2年ごとにLTSが出るので「**次の次のLTS**」まで使用できる
- LTSからLTSのアップグレードにも対応

Ubuntu Japanese Team

- Local Community Team（LoCo チーム）のひとつ
- Ubuntu がちょっと好きなただのボランティア集団
- 日本語ローカライズドイメージのリリース
- その他イベントの開催や紹介記事の執筆

Ubuntu と他の OS との違い

Ubuntu も中身は「Linux 系 OS」

- Ubuntu の運用で必要になる基礎知識
  - ▶ 他の Linux ディストリビューションと**大差ない**
- 特に最近はその差が小さくなってる
  - プログラミング言語ごとのパッケージ管理システム
  - コンテナによる隔離環境の活用
  - ▶ ホストとアプリの隔離技術の進化
- 気をつけるべき差は
  - 採用されているカーネル
  - 最初からインストールされているシステムソフトウェア
  - パッケージ管理システムとアップデート
  - 商用版でないと使えない機能がある
  - サポートなどソフトウェア以外の差
- つまり：インストール時がポイントになる

Ubuntu も中身は「Linux 系 OS」

- Ubuntu の運用で必要になる基礎知識
  - ▶ 他の Linux ディストリビューションと大差ない
- 特に最近はその差が小さくなってる
  - プログラミング言語ごとの**パッケージ管理システム**
  - **コンテナ**による隔離環境の活用
  - ▶ **ホスト**と**アプリ**の**隔離技術**の進化
- 気をつけるべき差は
  - 採用されているカーネル
  - 最初からインストールされているシステムソフトウェア
  - パッケージ管理システムとアップデート
  - 商用版でないと使えない機能がある
  - サポートなどソフトウェア以外の差
- つまり：インストール時がポイントになる

Ubuntu も中身は「Linux 系 OS」

- Ubuntu の運用で必要になる基礎知識
  - ▶ 他の Linux ディストリビューションと大差ない
- 特に最近はその差が小さくなってる
  - プログラミング言語ごとのパッケージ管理システム
  - **コンテナ**による隔離環境の活用
  - ▶ **ホスト**と**アプリ**の**隔離技術**の進化
- 気をつけるべき差は
  - 採用されているカーネル
  - 最初からインストールされているシステムソフトウェア
  - パッケージ管理システムとアップデート
  - 商用版でないと使えない機能がある
  - サポートなどソフトウェア以外の差
- つまり：インストール時がポイントになる

Ubuntu も中身は「Linux 系 OS」

- Ubuntu の運用で必要になる基礎知識
  ▶ 他の Linux ディストリビューションと大差ない
- 特に最近はその差が小さくなってる
  - プログラミング言語ごとのパッケージ管理システム
  - **コンテナ**による隔離環境の活用

  ▶ **ホスト**と**アプリ**の**隔離技術**の進化
- 気をつけるべき差は
  - 採用されているカーネル
  - 最初からインストールされているシステムソフトウェア
  - パッケージ管理システムとアップデート
  - 商用版でないと使えない機能がある
  - サポートなどソフトウェア以外の差
- つまり：**インストール時**がポイントになる

パッケージ管理システム

- パッケージ管理システムとパッケージフォーマットの例：
  - Ubuntu/Debian：**APT**（deb フォーマット）
  - RHEL/CentOS：**Yum/DNF**（RPM フォーマット）
  - SUSE/openSUSE：**Zypper**（RPM フォーマット）
- 基本的な使い方は**だいたい同じ**
- 細かく使い出すといろいろと差が出てくる
- pip や npm など言語ごとのパッケージと併用することも多い

RHEL/CentOS と比較したリポジトリの扱い

- RHEL のリポジトリ ≒Ubuntu の main/restricted
- EPEL≒Ubuntu の universe/multiverse
- Ubuntu は「**最初から EPEL が有効化済**」のような状態
- Canonical によるサポート対象に絞って言うと**ほぼ同じ**

リリース周期とサポート期間

- **Ubuntu の LTS：**
  - **2 年**ごとに**タイムベース**でリリース
  - **5 年間**のサポート、プラス 5 年の ESM（有償）[1]
- **RHEL：**
  - 約 3 年ぐらいごとに品質ベースでリリース
  - 約 10 年間のサポート、プラス 3 年の拡張サポート[2]
- **Debian：**
  - 約 2 年ぐらいごとに品質ベースでリリース
  - 約 3 年間のサポート、プラス 2 年の LTS[3][4]
- Ubuntu/Debian の**ポイントリリース**は機能追加がなく**保守的**
- RHEL の**マイナーリリース**はより積極的に機能が追加される[5]

---

[1] https://gihyo.jp/admin/clip/01/ubuntu-topics/201902/08

[2] https://access.redhat.com/support/policy/updates/errata

[3] https://wiki.debian.org/LTS/

[4] さらに ELTS（https://wiki.debian.org/LTS/Extended）という追加で 1 年以上のサポートを行う仕組みも存在します。

[5] より正確には機能追加は最初の 5 年に限定されます。

リリース周期とサポート期間

- **Ubuntu の LTS：**
  - **2 年**ごとに**タイムベース**でリリース
  - **5 年間**のサポート、プラス 5 年の ESM（有償）[1]
- **RHEL：**
  - **約 3 年**ぐらいごとに**品質ベース**でリリース
  - **約 10 年間**のサポート、プラス 3 年の拡張サポート[2]
- **Debian：**
  - 約 2 年ぐらいごとに品質ベースでリリース
  - 約 3 年間のサポート、プラス 2 年の LTS[3][4]
- Ubuntu/Debian の**ポイントリリース**は機能追加がなく**保守的**
- RHEL の**マイナーリリース**はより積極的に機能が追加される[5]

---

[1] https://gihyo.jp/admin/clip/01/ubuntu-topics/201902/08

[2] https://access.redhat.com/support/policy/updates/errata

[3] https://wiki.debian.org/LTS/

[4] さらに ELTS（https://wiki.debian.org/LTS/Extended）という追加で 1 年以上のサポートを行う仕組みも存在します。

[5] より正確には機能追加は最初の 5 年に限定されます。

リリース周期とサポート期間

- **Ubuntu の LTS：**
  - **2 年**ごとに**タイムベース**でリリース
  - **5 年間**のサポート、プラス 5 年の ESM（有償）[1]
- **RHEL：**
  - **約 3 年**ぐらいごとに**品質ベース**でリリース
  - **約 10 年間**のサポート、プラス 3 年の拡張サポート[2]
- **Debian：**
  - **約 2 年**ぐらいごとに**品質ベース**でリリース
  - **約 3 年間**のサポート、プラス 2 年の LTS[3][4]
- Ubuntu/Debian の**ポイントリリース**は機能追加がなく**保守的**
- RHEL の**マイナーリリース**はより積極的に機能が追加される[5]

---

[1] https://gihyo.jp/admin/clip/01/ubuntu-topics/201902/08

[2] https://access.redhat.com/support/policy/updates/errata

[3] https://wiki.debian.org/LTS/

[4] さらに ELTS（https://wiki.debian.org/LTS/Extended）という追加で 1 年以上のサポートを行う仕組みも存在します。

[5] より正確には機能追加は最初の 5 年に限定されます。

リリース周期とサポート期間

- **Ubuntu の LTS：**
  - **2 年**ごとに**タイムベース**でリリース
  - **5 年間**のサポート、プラス 5 年の ESM（有償）[1]
- **RHEL：**
  - **約 3 年**ぐらいごとに**品質ベース**でリリース
  - **約 10 年間**のサポート、プラス 3 年の拡張サポート[2]
- **Debian：**
  - **約 2 年**ぐらいごとに**品質ベース**でリリース
  - **約 3 年間**のサポート、プラス 2 年の LTS[3][4]
- Ubuntu/Debian の**ポイントリリース**は機能追加がなく**保守的**
- RHEL の**マイナーリリース**はより積極的に機能が追加される[5]

---

[1] https://gihyo.jp/admin/clip/01/ubuntu-topics/201902/08

[2] https://access.redhat.com/support/policy/updates/errata

[3] https://wiki.debian.org/LTS/

[4] さらに ELTS（https://wiki.debian.org/LTS/Extended）という追加で 1 年以上のサポートを行う仕組みも存在します。

[5] より正確には機能追加は最初の 5 年に限定されます。

リリース周期とサポート期間

- **Ubuntu の LTS：**
  - **2 年**ごとに**タイムベース**でリリース
  - **5 年間**のサポート、プラス 5 年の ESM（有償）[1]
- **RHEL：**
  - **約 3 年**ぐらいごとに**品質ベース**でリリース
  - **約 10 年間**のサポート、プラス 3 年の拡張サポート[2]
- **Debian：**
  - **約 2 年**ぐらいごとに**品質ベース**でリリース
  - **約 3 年間**のサポート、プラス 2 年の LTS[3][4]
- Ubuntu/Debian の**ポイントリリース**は機能追加がなく**保守的**
- RHEL の**マイナーリリース**はより積極的に機能が追加される[5]

---

[1] https://gihyo.jp/admin/clip/01/ubuntu-topics/201902/08

[2] https://access.redhat.com/support/policy/updates/errata

[3] https://wiki.debian.org/LTS/

[4] さらに ELTS（https://wiki.debian.org/LTS/Extended）という追加で 1 年以上のサポートを行う仕組みも存在します。

[5] より正確には機能追加は最初の 5 年に限定されます。

Debian との違い

- Debian は**サポートアーキテクチャ**が非常に多い
  - Ubuntu はたかだか 6 つ（amd64/i386/armhf/arm64/ppc64el/s390x）
  - Debian は公式サポートだけでも 10 個
非公式を含めるともっと多い[1]
- システムソフトウェアに対する選択肢の多さ
  - Ubuntu はシステムにとって重要なコンポーネントを決め打ちすることで、インストーラーの簡素化や短い周期でのリリースを実現している
  - Debian はできるだけユーザーが選択できるように注意深く作られている

[1] https://www.debian.org/ports/index.en.html

Debian との違い

- Debian は**サポートアーキテクチャ**が非常に多い
  - Ubuntu はたかだか 6 つ（amd64/i386/armhf/arm64/ppc64el/s390x）
  - Debian は公式サポートだけでも 10 個
非公式を含めるともっと多い[1]
- システムソフトウェアに対する選択肢の多さ
  - Ubuntu はシステムにとって**重要なコンポーネントを決め打ち**することで、インストーラーの簡素化や短い周期でのリリースを実現している
  - Debian はできるだけ**ユーザーが選択できる**ように注意深く作られている

[1] https://www.debian.org/ports/index.en.html

結論：基本的な差異は「慣れ」で解決

- どの OS もシステムを作っているのは**同じ人類**のはず
  - ターゲットユーザーもおそらく地球人
  - 人類に使いこなせないのならそれは**不具合**と言える
  - ただしユーザーが**本来の用途を誤解している**可能性はある
- 細かい部分になってくると**ケースバイケース**になる
- 普段から使い慣れているものを選ぶのがベスト
- あえて **Ubuntu を選ばなくても良い**状態が理想
- OS の選択権がないなら**全部使い慣れてしまおう！**

  ……もしくは出世しろ

結論：基本的な差異は「慣れ」で解決

- どの OS もシステムを作っているのは**同じ人類**のはず
  - ターゲットユーザーもおそらく地球人
  - 人類に使いこなせないのならそれは**不具合**と言える
  - ただしユーザーが**本来の用途を誤解している**可能性はある
- 細かい部分になってくると**ケースバイケース**になる
- 普段から使い慣れているものを選ぶのがベスト
- あえて **Ubuntu を選ばなくても良い**状態が理想
- OS の選択権がないなら**全部使い慣れてしまおう！**

  ……もしくは出世しろ

結論：基本的な差異は「慣れ」で解決

- どの OS もシステムを作っているのは**同じ人類**のはず
  - ターゲットユーザーもおそらく地球人
  - 人類に使いこなせないのならそれは**不具合**と言える
  - ただしユーザーが**本来の用途を誤解している**可能性はある
- 細かい部分になってくると**ケースバイケース**になる
- 普段から使い慣れているものを選ぶのがベスト
- あえて **Ubuntu を選ばなくても良い**状態が理想
- OS の選択権がないなら**全部使い慣れてしまおう！**

  ……もしくは出世しろ

結論：基本的な差異は「慣れ」で解決

- どの OS もシステムを作っているのは**同じ人類**のはず
  - ターゲットユーザーもおそらく地球人
  - 人類に使いこなせないのならそれは**不具合**と言える
  - ただしユーザーが**本来の用途を誤解している**可能性はある
- 細かい部分になってくると**ケースバイケース**になる
- 普段から使い慣れているものを選ぶのがベスト
- あえて **Ubuntu を選ばなくても良い**状態が理想
- OS の選択権がないなら**全部使い慣れてしまおう！**

  ……もしくは出世しろ

結論：基本的な差異は「慣れ」で解決

- どの OS もシステムを作っているのは**同じ人類**のはず
  - ターゲットユーザーもおそらく地球人
  - 人類に使いこなせないのならそれは**不具合**と言える
  - ただしユーザーが**本来の用途を誤解している**可能性はある
- 細かい部分になってくると**ケースバイケース**になる
- 普段から使い慣れているものを選ぶのがベスト
- あえて **Ubuntu を選ばなくても良い**状態が理想
- OS の選択権がないなら**全部使い慣れてしまおう！**

  ……もしくは出世しろ

ターゲットマシンとイメージの選択

どのリリースを使うのか

- **LTS**：2 年ごとにリリースされ **5 年間**のサポート
- **通常リリース**：半年ごとにリリースされ **9 ヶ月間**のサポート

どのリリースを使うのか

- **LTS**：2 年ごとにリリースされ **5 年間**のサポート
- **通常リリース**：半年ごとにリリースされ **9 ヶ月間**のサポート

サーバー用途なら LTS が無難

どうしても通常リリースが必要になるケース

- A. **最新のハードウェア**で動かしたい
- B. より**新しいカーネル**を使いたい
- C. より**新しいソフトウェア**を使いたい

A. 最新のハードウェアで動かしたい

- CPU やチップセット、GPU、NIC、ストレージ関連
- 発売されたばかりのデバイスはきちんと動かないことも
- カーネル・ユーザーランドのいずれかの更新が必要

ポイントリリースでも十分なことが多い

Ubuntu のポイントリリースについて

- 5 年の LTS 期間に登場する新ハードウェアも使いたい[1]
- **通常リリースのカーネル**を **LTS にバックポート**
- ファームウェア・グラフィックスタックも更新
- これらを ISO イメージにまとめて**半年に一度リリース**
- リリースされるイメージは**アップデート適用済み**
- 初回は LTS リリースの約 3 ヶ月後（アップデート適用のみ）
- 次の LTS まで **5 回**リリースする

[1] https://gihyo.jp/admin/serial/01/ubuntu-recipe/0278

Ubuntu 16.04 LTS の実績

- **2016 年 4 月 21 日：Ubuntu 16.04 LTS**
  Kernel 4.**4**、linux-firmware 1.157、xorg-xserver 1.18.3、Mesa 11.2.0
- **2016 年 7 月 21 日：Ubuntu 16.04.1 LTS**
  Kernel 4.4、linux-firmware 1.157**.2**、xorg-xserver 1.18.3、Mesa 11.2.0
- **2017 年 2 月 16 日：Ubuntu 16.04.2 LTS**
  Kernel 4.**8**、linux-firmware 1.157**.8**、xorg-xserver 1.18.**4**、Mesa **12.0.6**
- **2017 年 8 月 3 日：Ubuntu 16.04.3 LTS**
  Kernel 4.**10**、linux-firmware 1.157**.11**、xorg-xserver 1.**19.3**、Mesa **17.0.7**
- **2018 年 3 月 1 日：Ubuntu 16.04.4 LTS**
  Kernel 4.**13**、linux-firmware 1.157**.17**、xorg-xserver 1.19.**5**、Mesa 17.**2.8**
- **2018 年 8 月 2 日：Ubuntu 16.04.5 LTS**
  Kernel 4.**15**、linux-firmware 1.157**.20**、xorg-xserver 1.19.**6**、Mesa **18.0.5**

Ubuntu 18.04 LTS の実績

- **2018 年 4 月 26 日：Ubuntu 18.04 LTS**

  Kernel 4.**15**、linux-firmware 1.173、xorg-xserver 1.19.6、Mesa 18.0.5
- **2018 年 7 月 26 日：Ubuntu 18.04.1 LTS**

  Kernel 4.15、linux-firmware 1.173.**1**、xorg-xserver 1.19.6、Mesa 18.**2.2**
- **2019 年 2 月 15 日：Ubuntu 18.04.2 LTS**

  Kernel 4.**18**、linux-firmware 1.173.**3**、xorg-xserver 1.**20.1**、Mesa 18.2.**8**

B. より新しいカーネルを使いたい

- カーネルの特定の機能を使いたい
- パフォーマンスの都合
- ベンダーから特定のバージョンを指定された
- 不具合の調査

解決策

[1] https://gihyo.jp/admin/serial/01/ubuntu-recipe/0526
[2] https://wiki.ubuntu.com/Kernel/MainlineBuilds
[3] https://gihyo.jp/admin/serial/01/ubuntu-recipe/0524

解決策

- 最新の**ポイントリリース**のカーネルを使う
  - バージョンは指定できない。

[1] https://gihyo.jp/admin/serial/01/ubuntu-recipe/0526
[2] https://wiki.ubuntu.com/Kernel/MainlineBuilds
[3] https://gihyo.jp/admin/serial/01/ubuntu-recipe/0524

解決策

- 最新の**ポイントリリース**のカーネルを使う
  - バージョンは指定できない。
- カーネルのみ**自分でビルド**・インストールする
  - 複数台での再配布が面倒。

[1] https://gihyo.jp/admin/serial/01/ubuntu-recipe/0526
[2] https://wiki.ubuntu.com/Kernel/MainlineBuilds
[3] https://gihyo.jp/admin/serial/01/ubuntu-recipe/0524

解決策

- 最新の**ポイントリリース**のカーネルを使う
  - ▶ バージョンは指定できない。
- カーネルのみ**自分でビルド**・インストールする
  - ▶ 複数台での再配布が面倒。
- **独自のカーネルパッケージ**を作る

  「第 526 回 Ubuntu で最新のカーネルをお手軽にビルドする方法」[1]
  - ▶ セキュリティアップデートの追随が必要。

[1] https://gihyo.jp/admin/serial/01/ubuntu-recipe/0526
[2] https://wiki.ubuntu.com/Kernel/MainlineBuilds
[3] https://gihyo.jp/admin/serial/01/ubuntu-recipe/0524

解決策

- 最新の**ポイントリリース**のカーネルを使う
  - ▶ バージョンは指定できない。
- カーネルのみ**自分でビルド**・インストールする
  - ▶ 複数台での再配布が面倒。
- **独自のカーネルパッケージ**を作る

  「第 526 回 Ubuntu で最新のカーネルをお手軽にビルドする方法」[1]
  - ▶ セキュリティアップデートの追随が必要。
- **Mainline Builds** を使う（デバッグ用）[2]

  「第 524 回 Hades Canyon/Kaby Lake G の dGPU を有効化する」[3]
  - ▶「動くビルド」である保証はない。

[1] https://gihyo.jp/admin/serial/01/ubuntu-recipe/0526
[2] https://wiki.ubuntu.com/Kernel/MainlineBuilds
[3] https://gihyo.jp/admin/serial/01/ubuntu-recipe/0524

解決策

- 最新の**ポイントリリース**のカーネルを使う
  - ▶ バージョンは指定できない。
- カーネルのみ**自分でビルド**・インストールする
  - ▶ 複数台での再配布が面倒。
- **独自のカーネルパッケージ**を作る

  「第 526 回 Ubuntu で最新のカーネルをお手軽にビルドする方法」[1]
  - ▶ セキュリティアップデートの追随が必要。
- **Mainline Builds** を使う（デバッグ用）[2]

  「第 524 回 Hades Canyon/Kaby Lake G の dGPU を有効化する」[3]
  - ▶「動くビルド」である保証はない。

「独自パッケージ・ビルド」は
「独自の不具合」に遭遇するリスクがある

[1] https://gihyo.jp/admin/serial/01/ubuntu-recipe/0526
[2] https://wiki.ubuntu.com/Kernel/MainlineBuilds
[3] https://gihyo.jp/admin/serial/01/ubuntu-recipe/0524

C. より新しいソフトウェアを使いたい

- ライブラリのバージョンが古い
- 依存しているパッケージのバージョンが古い
- ソフトウェアに任意のパッチを適用したい

C. より新しいソフトウェアを使いたい

- ライブラリのバージョンが古い
- 依存しているパッケージのバージョンが古い
- ソフトウェアに任意のパッチを適用したい

ユーザーランドはコンテナに閉じ込めよう

ユーザーランドはコンテナに閉じ込めよう

- 必要なものが決まっている：**Docker**[1]
- いろいろなプロセスを動かしたい：**LXD**[2]
- 特定の言語のエコシステムを使う：**pyenv 他いろいろ**

[1] https://gihyo.jp/admin/serial/01/ubuntu-recipe/0458
[2] https://gihyo.jp/admin/serial/01/ubuntu-recipe/0521

ユーザーランドはコンテナに閉じ込めよう

- 必要なものが決まっている：**Docker**[1]
- いろいろなプロセスを動かしたい：**LXD**[2]
- 特定の言語のエコシステムを使う：**pyenv 他いろいろ**

直接**ホストにインストール**する
ソフトウェアは**可能な限り少なく**する

[1] https://gihyo.jp/admin/serial/01/ubuntu-recipe/0458
[2] https://gihyo.jp/admin/serial/01/ubuntu-recipe/0521

余談：デスクトップイメージは？

- インストール直後の**パッケージ構成が異なる**だけ
- 利用できるパッケージそのものに違いはない
- ubuntu-desktop：デスクトップ一式が入るメタパッケージ
- ubuntu-server：サーバー一式が入るメタパッケージ
- 当面の用途として GUI が欲しければデスクトップ版を選ぶ
- GUI は不要ならサーバー版を選ぶ

さまざまなインストール方法

何にインストールする？

- **物理マシン**：ISO イメージ、MAAS
- **クラウド**：AWS、Azure、GCE、Joyent、IBM Cloud
  - ▶ クラウドサービスごとの専用イメージが存在する[1]
- **コンテナ**：Docker、Kubernetes、LXD、Kata
- **VPS**：さくら、ConoHa
- **仮想マシン**：VirtualBox/Vagrant、VMWare、Hyper-V、virt-manager/Boxes、OpenStack、OpenNebula
  - ▶ 仮想マシン管理システムごとの専用イメージが存在する[1]
- **特殊**：Windows Subsystem for Linux（Bash on Windows）

[1] https://www.ubuntu.com/download/cloud
[2] https://cloud-images.ubuntu.com/releases/18.04/release/

どうやってインストールする？

- **Subiquity**：サーバー版のインストーラー
  - 手でポチポチ選んでいくので面倒
  - インストールを自動化しなくても良い場合に使う
  - 自動化のために preseed を使うなら旧インストーラーが必要
- cloud-init：クラウド/LXD イメージ向けの初期化ツール
  - YAML ファイルにインスタンスの初期状態を記述する
  - 大抵のクラウドサービスが対応している
  - Ubuntu のイメージなら仮想マシンでも利用可能
  - 本格的な構成管理をしたいなら Ansible などと併用する
- conjure-up/Juju：ソフトウェアスタック構築ツール[1]
  - 複数のマシン・インスタンスをセットアップし、個々のインスタンスに必要なソフトウェアを自動的にデプロイするツール
  - 初心者はまず気にしなくていい

[1] https://gihyo.jp/admin/serial/01/ubuntu-recipe/0469

どうやってインストールする？

- **Subiquity**：**サーバー版のインストーラー**
  - 手でポチポチ選んでいくので面倒
  - インストールを自動化しなくても良い場合に使う
  - 自動化のために preseed を使うなら旧インストーラーが必要
- **cloud-init**：**クラウド/LXD イメージ向けの初期化ツール**
  - YAML ファイルにインスタンスの初期状態を記述する
  - 大抵のクラウドサービスが対応している
  - Ubuntu のイメージなら仮想マシンでも利用可能
  - 本格的な構成管理をしたいなら Ansible などと併用する
- conjure-up/Juju：ソフトウェアスタック構築ツール[1]
  - 複数のマシン・インスタンスをセットアップし、個々のインスタンスに必要なソフトウェアを自動的にデプロイするツール
  - 初心者はまず気にしなくていい

[1] https://gihyo.jp/admin/serial/01/ubuntu-recipe/0469

どうやってインストールする？

- **Subiquity**：**サーバー版のインストーラー**
  - 手でポチポチ選んでいくので面倒
  - インストールを自動化しなくても良い場合に使う
  - 自動化のために preseed を使うなら旧インストーラーが必要
- **cloud-init**：**クラウド/LXD イメージ向けの初期化ツール**
  - YAML ファイルにインスタンスの初期状態を記述する
  - 大抵のクラウドサービスが対応している
  - Ubuntu のイメージなら仮想マシンでも利用可能
  - 本格的な構成管理をしたいなら Ansible などと併用する
- **conjure-up/Juju**：**ソフトウェアスタック構築ツール**[1]
  - 複数のマシン・インスタンスをセットアップし、個々のインスタンスに必要なソフトウェアを自動的にデプロイするツール
  - 初心者はまず気にしなくていい

[1] https://gihyo.jp/admin/serial/01/ubuntu-recipe/0469

ユースケースごとの提案

- 物理マシンに Ubuntu をインストール
- 仮想マシンに Ubuntu をインストール
- クラウドに Ubuntu をインストール
- ハイパーバイザーで他の OS と併用
- コンテナにインストール
- 10 台以上の Ubuntu をインストール・運用したい

物理マシンに Ubuntu をインストール

- 一番シンプルかつ確実
- オンプレミスで Ubuntu サーバーを運用する場合の選択肢
- アプリケーションは**コンテナに閉じ込める**とさらに良い
  - ホスト OS のアップグレード時に考慮すべきことが減る
  - 物理マシンのリプレース時にコンテナごと移動できる
- 電気代・場所・音・故障との戦いになる

仮想マシンに Ubuntu をインストール

- Windows/macOS 上で Ubuntu をお試し利用したい人向け
  - ▶ Windows/Hyper-V だと**数クリックでインストール完了**[1]
- **カーネル単位で隔離**したい人向け
  - ▶ 一般的なコンテナだとカーネルを共有してしまう
- cloud-init と組み合わせた**インストールの自動化**も可能
- Ubuntu サーバーの学習向けには十分
- コンテナに比べると**リソースが逼迫**しやすい
- 多インスタンスが必要な構成には不向き

[1] https://gihyo.jp/admin/serial/01/ubuntu-recipe/0549

クラウドに Ubuntu をインストール

- 世の中の大半の Ubuntu サーバーがこの形
- **即破棄・再インストールが容易**なので学習にも向いてる
- 複数のマシンを併用しやすい
- **cloud-init** と組み合わせるとさらに便利
- クラウドサービスに関する知識が必要（どうせこの先必須スキル）
- インターネットが必須（ギガが減る）

ハイパーバイザーで他の OS と併用

- 他の OS と共存・運用したい人向け
- ESXi とか Xen とか virt-manager とか
- ホストのリソースがそれなりに必要
- コンテナで十分なケースも増えてきた

コンテナにインストール

- すでにコンテナが動く環境があるならおすすめ
  - ホスト OS が Ubuntu である必要はない
  - **インストールの自動化**もかんたん
- **LXD**：システムコンテナ
  - ▶ Linux サーバーの勉強したいならおすすめ
- **Docker**：プロセスコンテナ
  - ▶ 個別のソフトウェアの運用を学ぶならこちらを使う
- Ubuntu なら LXD が最初からインストールされている
- **LXD の上で Docker を動かす**ことも可能

10 台以上の Ubuntu をインストール・運用したい

お前は聞きに来るセミナーを間違えた

気になる人は
OpenStack とか **Kubernetes** とか
そういう単語が入っているセミナーへ Go

Ubuntu サーバーの運用方法

管理者アカウントの扱い

- **root アカウント**は**ロック**されている
- 管理者権限が必要なら **sudo コマンドを利用する**
  sudo 実行したいコマンド
- どうしても root ログインしたい場合も sudo コマンドを使う
  sudo -i
- 管理者アカウントの考え方は Wiki 参照[1]

[1] https://wiki.ubuntulinux.jp/UbuntuTips/Others/RootSudo

パッケージのアップデート

- 手動でアップデートする場合は
  `sudo apt update && sudo apt upgrade`
- セキュリティアップデートは**自動で適用される**設定[1]
- 適用後に**システムの再起動**が必要な場合は通知される
- 主にカーネルのアップデート時に再起動する
- ライブラリの更新時はプロセスを個別に再起動する
- APT 関連の設定は apt_preferences(5) などを参照[2]

リリースの更新（LTS-to-LTS アップグレード）

- Ubuntu のリリース間の更新もコマンドラインで可能[3]
  `sudo apt update && sudo do-release-upgrade`

---

[1] 詳細は「unattended-upgrades」で検索

[2] http://manpages.ubuntu.com/manpages/bionic/ja/man5/apt_preferences.5.html

[3] https://help.ubuntu.com/community/Upgrades

Livepatch Service

- **再起動なし**に**カーネルパッチを適用する**仕組み
- 運用中のサービスを止めたくない場合に便利
- Canonical が提供する商用サービスだが無償で使うことも可能
- 詳しくは Ubuntu Weekly Recipe 第 443 回を参照[1]
  「再起動なしにカーネルを更新する『Canonical Livepatch Service』」

[1] https://gihyo.jp/admin/serial/01/ubuntu-recipe/0443

サポートの切れた環境を更新するには？

- **EOL（End of Life）**を迎えると**リポジトリが削除**される
- 削除されるとアップデート・アップグレードができなくなる
- 新しいリリースへは **EOL より前のアップグレード**が必須
- 移行し忘れた**うっかりさん**のために
  「**EOLUpgrades**」という仕組みも存在する[1]
- EOLUpgrades のリポジトリはあくまでアーカイブであり、
  セキュリティアップデートなどは提供されない

[1] https://help.ubuntu.com/community/EOLUpgrades

脆弱性情報の収集：Ubuntu Security Notices

- セキュリティアップデートに関する情報を公開するサービス[1]
- **ubuntu-security-announce**：上記のメール版[2]
- RSS フィードを Twitter に転送する非公式ボットも存在する[3]
- 週 1 程度の頻度で良ければ「Ubuntu Weekly Topics」もある[4]

[1] https://usn.ubuntu.com/
[2] https://lists.ubuntu.com/mailman/listinfo/ubuntu-security-announce
[3] https://twitter.com/ubuntu_advisory
[4] https://gihyo.jp/admin/clip/01/ubuntu-topics

ソフトウェアのインストール方法

- 公式リポジトリ
- PPA：Personal Package Archives
- サードパーティのリポジトリ
- snap リポジトリ
- その他の方法

公式リポジトリ

- インストールが簡単（apt コマンドや Ubuntu ソフトウェア）
- セキュリティアップデートが提供される
- バージョンが古くなりがち
- ビルド設定の変更に手間がかかる

```
$ apt search 検索ワード
$ sudo apt install パッケージ
```

ミラーリポジトリ

- 公式リポジトリには各地域ごとのミラーが用意されている
- 日本なら「jp.archive.ubuntu.com」
- クラウド・VPS が同様のミラーを用意していることもある
- 企業で大規模に使うなら独自ミラーを立てよう[1]

  第 315 回「apt-cacher-ng を使って APT 用キャッシュプロキシの構築」

[1] http://gihyo.jp/admin/serial/01/ubuntu-recipe/0315

PPA：Personal Package Archives

- Launchpad を利用した**独自リポジトリサービス**[1]
- ユーザーが自由にパッケージリポジトリを作れる
- **GitHub や各種 CI などとの連携**も可能
- 最新リリースやデイリービルドの提供に使われる

```
$ sudo add-apt-repository ppa:リポジトリ名
$ sudo apt install パッケージ名
```

[1] https://help.launchpad.net/Packaging/PPA

サードパーティのリポジトリ

- ベンダーが独自に構築した APT リポジトリ
- 登録が手作業なこと以外の使い方は PPA と同じ
- Docker ツールのインストールなどがこれに該当する

```
$ echo リポジトリ URI | sudo tee -a \
    /etc/apt/sources.list.d/リポジトリ名.list
$ sudo apt-key add リポジトリの鍵
$ sudo apt update
$ sudo apt install パッケージ名
```

snap リポジトリ

- ディストリビューションをまたいで利用できる**ユニバーサルパッケージシステム**
- 開発元が直接パッケージングすることを想定している
- 実行環境は**ホストから隔離**される
- Android アプリみたいなイメージ
- nodejs や Nextcloud は snap 版のほうが便利

```
$ snap find 検索ワード
$ sudo snap instal パッケージ
```

その他の方法

- **Docker コンテナ**
  - Docker イメージとしてインストール
  - Dockerfile から環境を構築
  - ソフトウェア提供元が Docker の利用を想定
- **Flatpak/AppImage**
  - snap と同じユニバーサルパッケージングシステム
  - Flatpak は Fedora で標準サポート？
- **Debian パッケージ**
  - バイナリパッケージを直接インストールする
  - `sudo apt instal ./パッケージ.deb`
  - rpm パッケージを deb ファイルに変換する方法も
- **ソースコード**
  - 昔ながらのソースコードからビルドする方法

どれを使えばいいの？

- バージョンに**制約がない**のであれば**公式リポジトリ**
- それ以外はソフトウェアの開発元のドキュメントに従う
- Ubuntu 上での管理の楽な順は次のとおり

  楽 公式リポジトリの apt/snap
  ↑ PPA・サードパーティリポジトリ
  ↓ Docker・Flatpak・AppImage・Debian パッケージ
  苦 ソースコード

- 「何をやったか」の記録を残すようにしよう
- その他のノウハウについては gihyo.jp 参照[1]
  第 331 回「パッケージ管理のハウツー集」

[1] https://gihyo.jp/admin/serial/01/ubuntu-recipe/0331

どれを使えばいいの？

- バージョンに制約がないのであれば公式リポジトリ
- それ以外はソフトウェアの**開発元のドキュメントに従う**
- Ubuntu 上での管理の楽な順は次のとおり

  楽 公式リポジトリの apt/snap

  ↑ PPA・サードパーティリポジトリ

  ↓ Docker・Flatpak・AppImage・Debian パッケージ

  苦 ソースコード

- 「何をやったか」の記録を残すようにしよう
- その他のノウハウについては gihyo.jp 参照[1]
  第 331 回「パッケージ管理のハウツー集」

[1] https://gihyo.jp/admin/serial/01/ubuntu-recipe/0331

どれを使えばいいの？

- バージョンに制約がないのであれば公式リポジトリ
- それ以外はソフトウェアの開発元のドキュメントに従う
- Ubuntu 上での管理の楽な順は次のとおり
  - **楽** 公式リポジトリの apt/snap
  - ↑ PPA・サードパーティリポジトリ
  - ↓ Docker・Flatpak・AppImage・Debian パッケージ
  - **辛** ソースコード
- 「何をやったか」の記録を残すようにしよう
- その他のノウハウについては gihyo.jp 参照[1]
  第 331 回「パッケージ管理のハウツー集」

[1] https://gihyo.jp/admin/serial/01/ubuntu-recipe/0331

どれを使えばいいの？

- バージョンに制約がないのであれば公式リポジトリ
- それ以外はソフトウェアの開発元のドキュメントに従う
- Ubuntu 上での管理の楽な順は次のとおり
  - 楽 公式リポジトリの apt/snap
  - ↑ PPA・サードパーティリポジトリ
  - ↓ Docker・Flatpak・AppImage・Debian パッケージ
  - 辛 ソースコード
- 「**何をやったか**」の**記録を残す**ようにしよう
- その他のノウハウについては gihyo.jp 参照[1]
  第 331 回「パッケージ管理のハウツー集」

[1] https://gihyo.jp/admin/serial/01/ubuntu-recipe/0331

どれを使えばいいの？

- バージョンに制約がないのであれば公式リポジトリ
- それ以外はソフトウェアの開発元のドキュメントに従う
- Ubuntu 上での管理の楽な順は次のとおり
  - **楽** 公式リポジトリの apt/snap
  - ↑ PPA・サードパーティリポジトリ
  - ↓ Docker・Flatpak・AppImage・Debian パッケージ
  - **辛** ソースコード
- 「何をやったか」の記録を残すようにしよう
- その他のノウハウについては gihyo.jp 参照[1]
  第 331 回「パッケージ管理のハウツー集」

[1] https://gihyo.jp/admin/serial/01/ubuntu-recipe/0331

その他の知っておくべき基本ツール

- サービス管理：systemd/systemd-journald
- ネットワーク設定：netplan/systemd-networkd
- ファイヤーウォール：ufw/iptables
- 設定ファイルの編集：nano/vim
- 端末：tmux/byobu
- Landscape：Canonical 製の商用管理ツール[1]

[1] https://landscape.canonical.com/

Ubuntu サーバーガイド

https://help.ubuntu.com/lts/serverguide/index.html.ja

一通り読んでおくと
「**Ubuntu はこういう時こうする**」
というのがざっと理解できる。

Ubuntu に関する日本語の情報源

Ubuntu Weekly Topics/Recipe

- https://gihyo.jp/admin/clip/01/ubuntu-topics
- https://gihyo.jp/admin/serial/01/ubuntu-recipe
- **Ubuntu の最新情報や便利な使い方**などをお届け

Software Design：Ubuntu Monthly Report

- https://gihyo.jp/magazine/SD/
- Ubuntu Japanese Team で連載中
- 2019 年 3 月号はいくやさんによる
  「第 106 回 LibreOffice 6.2 の新機能」

日経 Linux

- http://trendy.nikkeibp.co.jp/linux/
- 水野さんが「**Linux 100 %活用ガイド**」を連載中
- 長南さんが「**Linux 12 星座占い**」と RasPi 関連を連載中
- 2019 年 3 月号には
  「**Ubuntu 18.10 日本語 Remix**」
  「**Xubuntu 18.04 LTS**」
  のブータブルディスクなどが付属
- **Linux 入門**とか**古い PC を Linux で**なんて記事も

Ubuntu 18.04 LTS 関連本

- 「**Ubuntu スタートアップバイブル**」小林準[1]
- 「**Ubuntu サーバー徹底入門**」中島能和[2]
- 「**Ubuntu 18.04 LTS 日本語 Remix 使い方が全部わかる本**」日経 Linux 編集部[3]
- 「**Ubuntu はじめる&楽しむ 100%活用ガイド**」リンクアップ[4]

[1] https://book.mynavi.jp/ec/products/detail/id=92010
[2] https://www.shoeisha.co.jp/book/detail/9784798155760
[3] https://www.nikkeibp.co.jp/atclpubmkt/book/18/270650/
[4] https://gihyo.jp/book/2018/978-4-297-10018-6

うぶんちゅ！ まがじん ざっぱ〜ん♪

- http://zapppaaan.freepub.jp/
- 商業誌には書きづらいあれやこれやのネタ
- DRM なしの電子版（EPUB + PDF）による販売
- 6 月に公開された最新号の vol.8 では 18.04 を特集
- **技術書典 6**[1]に出展決定！

[1] https://techbookfest.org/event/tbf06

技術書典 6 のおしながき

- うぶんちゅ！ まがじん ざっぱ〜ん♪ vol.9
- ざっクリわかる Ubuntu 18.04 LTS Server

Ubuntu オフラインミーティング

- Ubuntu を肴に飲み食いするイベント
- Ubuntu 19.04 がリリースされたらまたなんかやります
- ML：
  https://lists.ubuntu.com/mailman/listinfo/ubuntu-jp
- 18.04 の時のフォトレポート：
  https://gihyo.jp/admin/serial/01/ubuntu-recipe/0525
- 18.10 の時のフォトレポート：
  https://gihyo.jp/admin/serial/01/ubuntu-recipe/0550