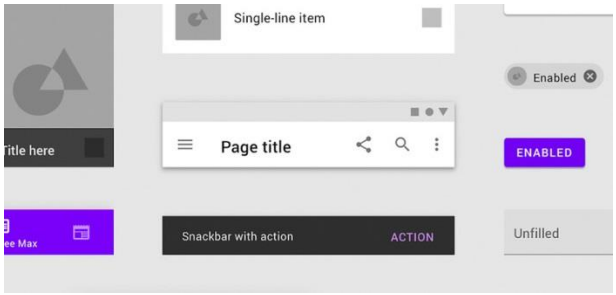
4 Material Design

Material Design – це дизайн-система, яку винайшли та представили в компанії Google у 2014 році. Це не просто гайдлайн за єдиним візуальним оформленням – завдяки йому були уніфіковані інтерфейси всіх продуктів та сервісів корпорації, зокрема ОС Android. В результаті їх сукупність сприймається як єдина цифрова система, створюючи таким чином новий досвід користувача і забезпечуючи проникнення сервісів корпорації у всі сфери життя людини.

До впровадження цієї концепції не те щоб у різних продуктах, а навіть у різних версіях одного сервісу (у мобільній та десктопній) дизайн значно відрізнявся. Через це користувачам було складно орієнтуватися під час переходу між ними. Навіть спроба розробити єдиний стиль для Android Holo не вирішила проблему – люди все одно плуталися в інтерфейсах. Лише з переходом до принципу Material Design користувальницький досвід вдалося покращити.



4.1 Навіщо він потрібен

Як вже було раніше сказано, Material Design призначений для уніфікації інтерфейсів у продуктах та сервісах компанії. На думку розробників Google, об'єкти інтерфейсу повинні мати аналог, метафору в реальному світі. І як метафора вони взяли звичайний папір і чорнило. За рахунок цього було створено асоціацію з реальним світом та використанням знайомих тактильних характеристик, глибини.

Папір в Material Design, точно як у реальному світі, характеризується тонкістю і площиною, також має тінь. Крім нормальних фізичних якостей у ній також є певна частка «чарівництва» – анімація. Завдяки такому рішенню користувачеві легше зрозуміти принципи роботи ПЗ та переходи від одного стану до іншого. Виходить, що анімація і пожвавлює інтерфейс, і показує користувачеві, як усе працює.

4.2 Основні принципи Material Design

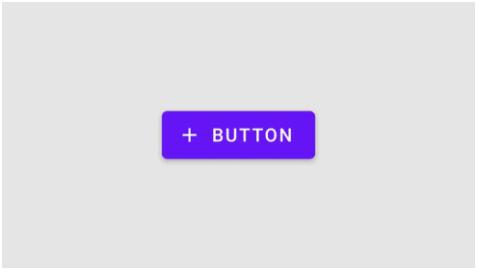
В основі концепції Material Design лежать 4 принципи:

Тактильні поверхні . Основою формування простору є «цифровий папір». Окремі її листи розташовуються на різній висоті і при цьому відкидають тіні один на одного. Крім іншого, вони можуть розтягуватися, змінювати форму і колір, і з'єднуватися друг з одним. За рахунок цього користувачі краще розуміють, як влаштована система та яка її ієрархія.

Поліграфічний дизайн. Це означає, що при створенні інтерфейсу цифрових пристроїв використовуються традиційні засоби та підходи з графічного дизайну. Скажімо так, на «цифровому папері» елементи виводяться за допомогою «цифрового чорнила».

Осмислена анімація. Тут анімація не просто вискакує з нізвідки, а з'являється відповідно до логіки системи. Тобто один об'єкт, реагуючи на дії користувача, плавно перетворюється на інший.

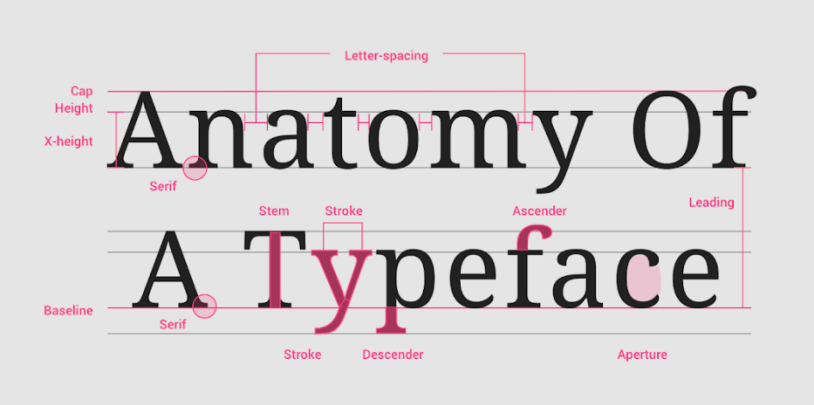
Адаптивний дизайн. Тут все просто - інтерфейс повинен бути оптимізований на всіх пристроях та екранах незалежно від того, який продукт ми беремо як приклад. При цьому виглядати, працювати та реагувати все має скрізь однаково.



4.3 Інші особливості

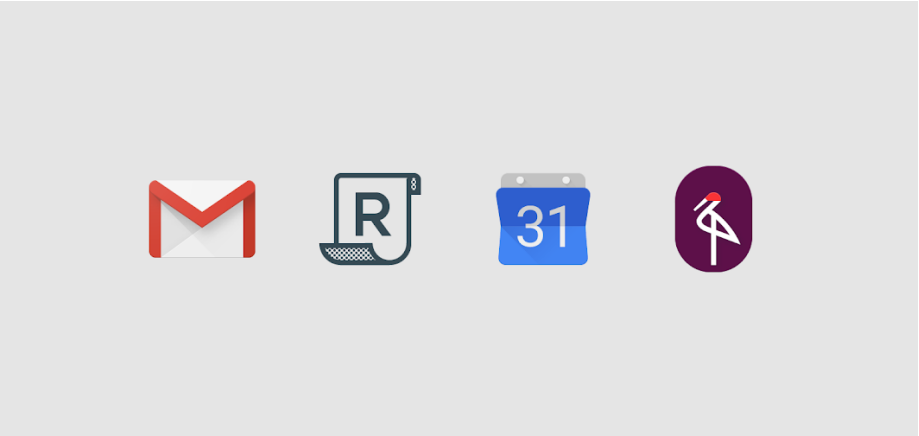
Вони випливають із основних принципів Material Design і полягають у наступному:

- **Наявність глибини тіней.** Воно надає обсягу звичайному плоскому дизайну і тим самим задає певний функціонал - позначення структури елементів. Наприклад, що вище підйом об'єкта, то більше вписувалося тінь.
- **Контрастна друкарня.** Сам собою типографіка задає стиль бренду і створює структуру контенту. У другому випадку це добре видно на прикладі заголовків - вони виділяються більшим і темнішим шрифтом. Це дозволяє створити контраст між заголовками та набірним текстом, що і функціонально, і красиво.



- **Модульна сітка.** Ця техніка прийшла із поліграфічного дизайну. Деталі розташовуються відповідно до ключових напрямних. Сітка створює відступи та краще демонструє структурованість інформації.
- **Яскраві кольори.** У цій концепції існують основні та акцентні кольори. Ціль основного кольору - позначити великі області. Акцентний трохи яскравіший, використовується точково і в невеликій кількості для виділення елементів управління, таких як кнопки, індикатори і так далі. Тим самим він дозволяє привертати увагу користувача до ключових деталей.
- **Реакція анімації.** Будь-який об'єкт повинен реагувати виключно на дії користувача, будь то торкання пальцем на мобільних пристроях або наведення курсору на десктопних версіях.

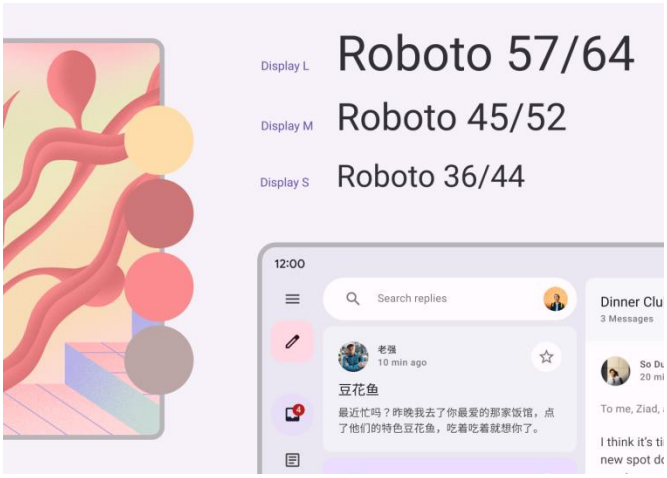
- **Від загального до часткового.** Цей принцип полягає в зменшенні кількості інформації, що подається, зі зменшенням розміру екрана. Відповідно, на великих дисплеях можна показати одразу і списки, і детальніші відомості. А ось на маленьких екранах для цього можна зробити список, що розкривається, або навіть перекласти подробиці на новий екран.



4.4 Користь Material Design

У Google при створенні Material Design орієнтувалися на досвід користувача і функціональність деталей системи. У результаті інтерфейс стає простішим і виразнішим. Саме анімація є основним елементом системи, завдяки якому робляться такі акценти:

1. **Демонстрація взаємозв'язків.** Завдяки анімації стає простою і зрозумілою ієрархія, людина легко зможе зрозуміти, що станеться при натисканні на конкретну деталь.
2. **Фокусування уваги.** Анімація в жодному разі не відволікає від основного впливу, що відповідає третьому принципу – її усвідомленості.
3. **Виразність.** Кожен продукт компанії має свій особливий характер, стиль і все це чітко відображено в анімованих деталях.
4. **Навчання.** Люди без особливих зусиль можуть зрозуміти, як виконувати ті чи інші дії.
5. **привабливість.** Навіть якщо виключити практичну складову, то вся анімація сама собою виглядає досить цікаво. У результаті користувач виявляє інтерес і хоче взаємодіяти із продуктом.



Material Design 3 для Android 12

Завдяки концепції Material Design багато компаній переглянули підходи до створення дизайнерських елементів. Зокрема, анімація з доповнення перетворилася на чи не повноцінну частину інтерфейсу. При цьому на чолі всього цього було закладено саме досвід користувача, його оптимізація і спрощення. Цей підхід став у результаті популярним і поставив напрямок для розвитку всього цифрового дизайну.

4.5 Components для Android

Material Components для Android доступні через репозиторій Google Maven. Щоб використовувати:

Відкрийте `build.gradle` файл для вашої програми.

Переконайтеся, що `сховища` містить репозиторій Google Maven `google()`. Наприклад:

```
allprojects {
  repositories {
    google()
    jcenter()
  }
}
```

Додайте бібліотеку до `залежності` розділ:

```
dependencies {
  // ...
  implementation 'com.google.android.material:material:<version>'
  // ...
}
```

Відвідайте Репозиторій Google Maven або Репозиторій MVN щоб знайти останню версію бібліотеки.

Новий простір імен і AndroidX

Якщо ваша програма зараз залежить від оригінальної бібліотеки підтримки дизайну, ви можете скористатися `Рефакторинг до AndroidX…` варіант, наданий Android Studio. Це оновить залежності вашої програми та код для використання нової упаковки`androidx`і`com.google.android.material` бібліотеки.

Якщо ви не хочете переходити на новий `androidx`і `com.google.android.material` пакунків, ви можете використовувати матеріальні компоненти через `com.android.support:design:28.0.0` залежність.

Примітка. Ви не повинні використовувати`com.android.support`і `com.google.android.material`залежностей у вашій програмі одночасно.

2. Скомпілюйте свою програму з Android 10

Щоб використовувати Material Components для Android і останні версії бібліотек підтримки, вам потрібно буде встановити Android Studio 3.5 або новішої версії для створення з Android 10 і оновити програму`compileSdkVersion`до`29`.

3. Переконайтеся, що ви використовуєте`AppCompatActivity`

Використання`AppCompatActivity`забезпечить правильну роботу всіх компонентів. Якщо ви не можете продовжити з`AppCompatActivity`, оновіть свою діяльність, щоб використовувати `AppCompatDelegate`. Це дозволить`AppCompat`версії компонентів, які потрібно роздути серед інших важливих речей.

4. Змініть тему програми, щоб успадкувати тему Material Components

Рекомендовано виконати міграцію всієї програми, змінивши тему програми на спадкування теми Material Components. Однак після цього обов’язково ретельно перевірте, оскільки компоненти в існуючих макетах можуть змінити свій вигляд і поведінку.

Примітка. Якщо ви **не можете** змінити тему, ви можете зробити одне з наведеного нижче.

Успадкуйте одну з наших тем Material Components **Bridge** . Див **Теми Bridge**розділ для отримання додаткової інформації.

Продовжуйте успадковувати тему AppCompat і додайте до неї деякі нові атрибути теми. Див **Теми AppCompat** розділ для отримання додаткової інформації.

Теми Матеріальні компоненти

Нижче наведено список тем матеріальних компонентів, які можна використовувати, щоб отримати найновіші стилі компонентів і атрибути рівня теми.

`Theme.MaterialComponents`
`Theme.MaterialComponents.NoActionBar`
`Theme.MaterialComponents.Light`
`Theme.MaterialComponents.Light.NoActionBar`
`Theme.MaterialComponents.Light.DarkActionBar`
`Theme.MaterialComponents.DayNight`
`Theme.MaterialComponents.DayNight.NoActionBar`
`Theme.MaterialComponents.DayNight.DarkActionBar`

Оновіть тему програми, щоб успадкувати одну з цих тем, наприклад:

```
<style name="Theme.MyApp" parent="Theme.MaterialComponents.DayNight">
    <!-- ... -->
</style>
```

Щоб дізнатися більше про те, як налаштувати атрибути рівня теми для вашої програми, перегляньте нашТематизаціякерівництво, а також наш Темна темадовідник про те, чому важливо успадковувати від `День ніч`тема.

Примітка. Використання теми Material Components увімкне користувальницький розширювач перегляду, який замінює компоненти за замовчуванням їхніми матеріальними аналогами. Наразі це лише заміна`< Кнопка >`і`< AutoCompleteTextView >`Компоненти XML з `< MaterialButton >`і `< MaterialAutoCompleteTextView >`, відповідно.

Теми Bridge

Якщо ви не можете змінити свою тему, щоб успадкувати її від теми Material Components, ви можете успадкувати тему Material Components **Bridge** .

```
<style name="Theme.MyApp" parent="Theme.MaterialComponents.Light.Bridge">
    <!-- ... -->
</style>
```

- `Theme.MaterialComponents.Bridge`
- `Theme.MaterialComponents.Light.Bridge`
- `Theme.MaterialComponents.NoActionBar.Bridge`
- `Theme.MaterialComponents.Light.NoActionBar.Bridge`

- `Theme.MaterialComponents.Light.DarkActionBar.Bridge`

Теми Bridge успадковують теми AppCompat, але також визначають для вас нові атрибути теми Material Components. Якщо ви використовуєте тему мосту, ви можете почати використовувати компоненти Material Design, не змінюючи тему програми.

Теми AppCompat

Ви також можете поступово тестувати нові компоненти Material, не змінюючи тему програми. Це дозволяє зберегти ваші існуючі макети в незмінному вигляді та повести себе, вводячи нові компоненти у ваш макет по черзі.

Однак ви повинні додати наступні нові атрибути теми до вашої існуючої теми програми, інакше ви зіткнетесяThemeEnforcementпомилки:

```
<style name="Theme.MyApp" parent="Theme.AppCompat">

  <!-- Original AppCompat attributes. -->
  <item name="colorPrimary">@color/my_app_primary_color</item>
  <item name="colorSecondary">@color/my_app_secondary_color</item>
  <item name="android:colorBackground">@color/my_app_background_color</item>
  <item name="colorError">@color/my_app_error_color</item>

  <!-- New MaterialComponents attributes. -->
  <item name="colorPrimaryVariant">@color/my_app_primary_variant_color</item>
  <item
name="colorSecondaryVariant">@color/my_app_secondary_variant_color</item>
  <item name="colorSurface">@color/my_app_surface_color</item>
  <item name="colorOnPrimary">@color/my_app_color_on_primary</item>
  <item name="colorOnSecondary">@color/my_app_color_on_secondary</item>
  <item name="colorOnBackground">@color/my_app_color_on_background</item>
  <item name="colorOnError">@color/my_app_color_on_error</item>
  <item name="colorOnSurface">@color/my_app_color_on_surface</item>
  <item name="scrimBackground">@color/mtrl_scrim_color</item>
  <item
name="textAppearanceHeadline1">@style/TextAppearance.MaterialComponents.Headli
ne1</item>
  <item
name="textAppearanceHeadline2">@style/TextAppearance.MaterialComponents.Headli
ne2</item>
  <item
name="textAppearanceHeadline3">@style/TextAppearance.MaterialComponents.Headli
ne3</item>
  <item
name="textAppearanceHeadline4">@style/TextAppearance.MaterialComponents.Headli
ne4</item>
  <item
name="textAppearanceHeadline5">@style/TextAppearance.MaterialComponents.Headli
ne5</item>
  <item
name="textAppearanceHeadline6">@style/TextAppearance.MaterialComponents.Headli
ne6</item>
```

```
  <item
name="textAppearanceSubtitle1">@style/TextAppearance.MaterialComponents.Subtit
le1</item>
  <item
name="textAppearanceSubtitle2">@style/TextAppearance.MaterialComponents.Subtit
le2</item>
  <item
name="textAppearanceBody1">@style/TextAppearance.MaterialComponents.Body1</ite
m>
  <item
name="textAppearanceBody2">@style/TextAppearance.MaterialComponents.Body2</ite
m>
  <item
name="textAppearanceCaption">@style/TextAppearance.MaterialComponents.Caption<
/item>
  <item
name="textAppearanceButton">@style/TextAppearance.MaterialComponents.Button</i
tem>
  <item
name="textAppearanceOverline">@style/TextAppearance.MaterialComponents.Overlin
e</item>

</style>
```

Реалізація текстового поля через XML

За замовчуванням заповнене текстове поле XML визначається як:

```
<com.google.android.material.textfield.TextInputLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:hint="@string/textfield_label">

  <com.google.android.material.textfield.TextInputEditText
      android:layout_width="match_parent"
      android:layout_height="wrap_content"/>
</com.google.android.material.textfield.TextInputLayout>
```

Примітка: якщо ви **не** використовуєте тему, яка успадковує тему Material Components, вам також доведеться вказати стиль текстового поля за допомогою `style="@style/Widget.MaterialComponents.TextInputLayout.FilledBox"`

Також надаються інші стилі текстових полів. Наприклад, якщо ви хочете окреслене текстове поле у вашому макеті ви можете застосувати матеріальні компоненти `окреслено` стиль для текстового поля в XML:

```
<com.google.android.material.textfield.TextInputLayout
    style="@st
```