**onchange**

* The @api.onchange decorator is used to trigger a method when a specified field's value is changed in the UI.
* It is only used for transient UI changes and does not store the result in the database.
* Works only in the form view and does not affect computed values when records are created/updated via code (ORM, API, etc.).
* **Stored**: No, because it does not write to the database.
* **Editable**: Yes, because it works in UI.

**Example:**

from odoo import models, fields, api

class ProductTemplate(models.Model):

\_inherit = 'product.template'

cost\_price = fields.Float(string="Cost Price")

sale\_price = fields.Float(string="Sale Price")

@api.onchange('cost\_price')

def \_onchange\_cost\_price(self):

if self.cost\_price:

self.sale\_price = self.cost\_price \* 1.2 # 20% margin

* Here, when cost\_price changes in the UI, sale\_price is automatically updated.

**depends**

* The @api.depends decorator is used for computed fields to track dependencies and automatically update their values.
* **Stored**: It can be stored if **store=True** is set, otherwise, it is calculated dynamically.
* **Editable**: No, because it is always computed.
* **Readonly**: Yes, unless overridden by another mechanism.

**Example:**

class ProductTemplate(models.Model):

\_inherit = 'product.template'

cost\_price = fields.Float(string="Cost Price")

sale\_price = fields.Float(string="Sale Price", compute="\_compute\_sale\_price", store=True)

@api.depends('cost\_price')

def \_compute\_sale\_price(self):

for record in self:

record.sale\_price = record.cost\_price \* 1.2 # 20% margin

* Here, sale\_price is **computed** whenever cost\_price changes.
* Since store=True, it is stored in the database and can be used in searches.

**Key Differences Between @api.onchange and @api.depends**

| **Feature** | **@api.onchange** | **@api.depends** |
| --- | --- | --- |
| **Triggers in UI?** | Yes | No |
| **Triggers in ORM/API?** | No | Yes |
| **Stored in DB?** | No | Yes (if store=True) |
| **Editable?** | Yes | No |
| **Readonly?** | No | Yes (by default) |

**Can @api.depends Fields Be Editable?**

* **No**, because computed fields are readonly by default.
* However, if you set store=True and readonly=False, you can make it editable manually.
* **But generally, computed fields should not be edited manually.**

**Example:**

class ProductTemplate(models.Model):

\_inherit = 'product.template'

total\_price = fields.Float(string="Total Price", compute="\_compute\_total\_price", store=True, readonly=False)

@api.depends('cost\_price')

def \_compute\_total\_price(self):

for record in self:

record.total\_price = record.cost\_price \* 1.2

**Comparison Table: onchange vs. depends**

| **Feature** | **Onchange** | **Depends** |
| --- | --- | --- |
| **Trigger** | Field change in the form view. | Field change in the database or UI. |
| **Scope** | Only updates the form view (temporary). | Updates both UI and database (permanent). |
| **Save** | Does **not save** data to the database automatically.  If use store data use **write method**. | Saves data to the database if **store=True**. |
| **Usage Context** | User interaction in forms. | Backend logic, reporting, or automated processes. |
| **Fields** | Works with any field type. | Works with computed fields. |
| **Views Supported** | **Form view only**. | **All views** (form, tree, kanban, etc.). |
| **Dependencies** | Defined directly in the decorator (onchange). | Declared in @api.depends. |
| **Method Decorator** | @api.onchange('field\_name') | @api.depends('field\_name') |