|  |
| --- |
|  |
| Introduction à VBA Access Microsoft |
|  |

|  |
| --- |
| 11/01/2025 |

Qu’est-ce que VBA ?

L

e VBA (Visual Basic for Applications) est un langage de programmation intégré à Microsoft Access, qui permet d’automatiser des tâches, de personnaliser des fonctionnalités et de développer des applications personnalisées. Dans Access, VBA est particulièrement utile pour manipuler les données, créer des interfaces utilisateur dynamiques, et effectuer des actions qui ne sont pas possibles avec les macros ou les fonctions standards.

# Quelques actions possibles avec VBA Access

**1. Manipulation des objets Access:**

* Ouvrir, fermer et modifier des formulaires, états, tables et requêtes.
* Ajouter ou supprimer des enregistrements dans une table.

**2. Gestion des événements :**

* Réagir aux actions des utilisateurs, comme le clic sur un bouton ou la sélection d'une valeur dans une liste déroulante.
* Valider les entrées utilisateur avant qu'elles ne soient enregistrées dans la base de données.

**2. Automatisation des tâches :**

* Lancer des requêtes (SELECT, INSERT, UPDATE, DELETE) directement via VBA.
* Importer et exporter des données dans différents formats (Excel, CSV, XML).
* Générer et imprimer des rapports à la demande.

**4. Interopérabilité :**

* Lire et écrire dans des fichiers Excel ou Word.
* Envoyer des emails automatiquement via Outlook.
* Connecter Access à des bases de données externes (SQL Server, MySQL).

**5. Création de fonctionnalités avancées :**

* Ajouter des boutons avec des actions personnalisées.
* Développer des modules réutilisables pour des fonctionnalités spécifiques.
* Sécuriser les données en implémentant des systèmes de permissions avancés.

**6. Gestion des erreurs et logs :**

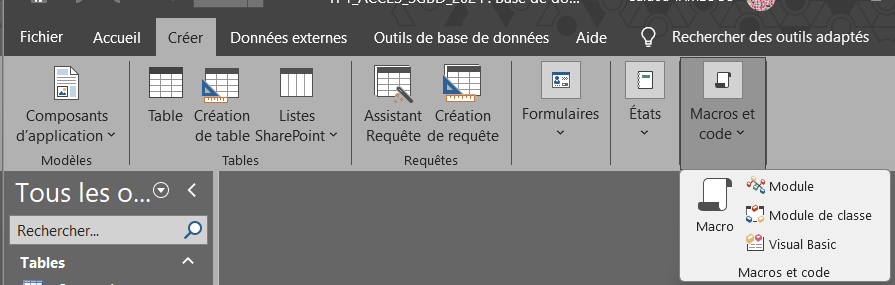
* Capturer et traiter les erreurs pour éviter que l'application ne plante.
* Ajouter des journaux d’audit pour suivre les modifications des données

# Editeur de code VBA Access

## Ouverture de l’éditeur VBA Access

Cliquer sur l'**onglet Créer** en haut de la fenêtre **Access**, pour activer son ruban,

📌 Dans la section Macros et code, tout à fait à droite du ruban, cliquer sur le bouton **Visual Basic**,

****

📌 Si un formulaire est déjà ouvert en mode création, clicker sur « **Visualiser le code** »

📌 Le raccourci clavier **Alt + F11** peut être utiliser pour basculer de la fenêtre de l'application à celle de l'éditeur.

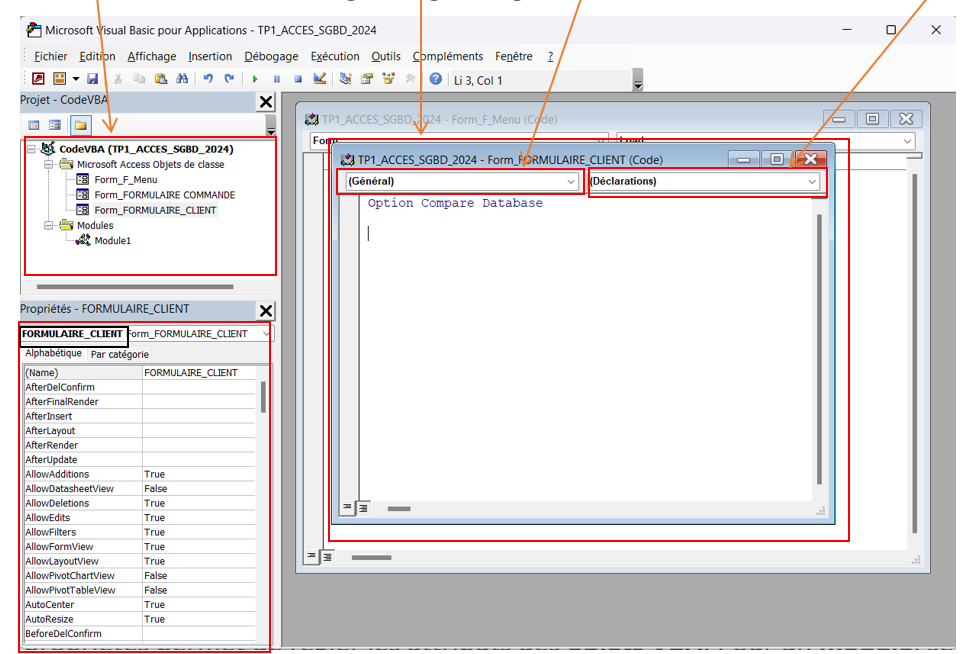
Nous basculons ainsi dans l'**éditeur de code Visual Basic pour Access**

### Liste des actions

### Explorateur du projet : les éléments sont regroupés dans des modules. On peut renommer ces éléments.

### Liste deroulante regroupant toutes les commandes des objets du formulaire actif. Par exemple command1

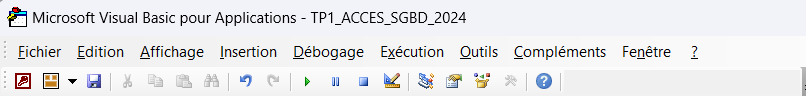
### L’interface de saisie du code



### Espace de renommage des objets comme les modules

### Cette fenêtre propriétés permet de régler les attributs des **objets VBA**. Pour un **module**, seule la **propriété (Name)** est concernée

## Les onglets de l’éditeur VBA

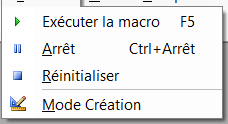


-Enregistrer

### -importer ou exporter un fichier

-imprimer

-fermer



-inserer un module

### - inserer une procedure

-inserer un module de classe

# Création d’une commande en VBA

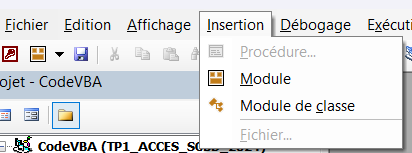
## Créer un module

Un module permet de regrouper plusieurs macros en VBA.

⚠️ Tout code VBA doit être écrit dans un module.

Nous devons donc en ajouter un.

📌 Cliquer sur le menu **Insertion** en haut de l'éditeur et choisir Module dans la liste,

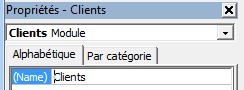


Un nouveau **Module** nommé Module1 par défaut, est ajouté. Il apparaît dans l'**explorateur de projet** situé, en haut à gauche de l'**éditeur de code**. Ce nom peut être modifié grâce à la fenêtre **Propriétés** qui s'affiche en bas à gauche de l'**éditeur de code**.

## Pour renommer le module actif

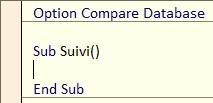
✍️ Dans le volet de gauche, cliquer dans la case *(Name)*

✍️ Saisir un nom pour le module (exemple : Clients)



## CRÉATION D’UNE PROCÉDURE VBA

La procédure va contenir les différentes commandes de la macro en VBA



* 1. Menu Insertion Procédure

|  |
| --- |
| **OK** |

* 1. Saisir le nom de la procédure voulue (exemple : Suivi) sans espace Laisser *Sub* cocher et

❗ Il est possible de saisir directement au clavier le titre de sa macro dans son module,

Exemple : *Sub Suivi*, les parenthèses et le *End Sub* sont ajoutés automatiquement.

## SASIE DES COMMANDES VBA

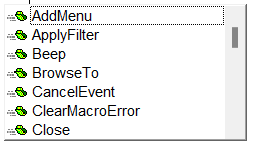
Entre le *Sub* et *End Sub*, saisir le code — l’ensemble des commandes — de sa macro :

* *Option Compare Database* est une commande facultative indiquant que les tris et les comparaisons des textes seront faits en fonction des paramètres régionaux de la base de données
* *Private* devant une procédure indique que celle‐ci n’est accessible qu’à partir de son module, exemple : Private Sub Suivi() qui est la valeur par défaut.
* *Public* indique que la procédure est accessible à partir de tous les modules, exemple : Public Sub Suivi()

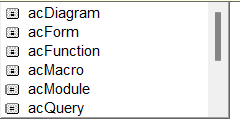
## Syntaxe générale

**DoCmd**. « Nom de la fonction ou methode » « le type de l’objet sur lequel l’action est faite » « ‘’Nom de l’objet’’ » « [autres options] »

* + Quelques méthodes :



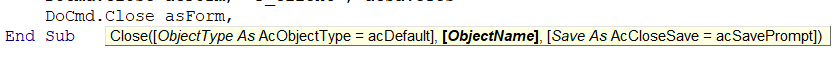
* + Quelques types d’objets



✍️ **Astuces**

Vous n’avez pas à mémoriser ces éléments mais comprendre leurs utilisations. Lors de la saisie, Access fait des autocompletions ( affiche les éléments qui peuvent être utiliser lorsque vous mettez un « point » ou « virgule »).

Access nous guide lors de l’utilisation d’une méthode par exemple :



# QUELQUES EXEMPLES DE CODE VBA SUR LES FORMULAIRES

## OUVRIR UN FORMULAIRE

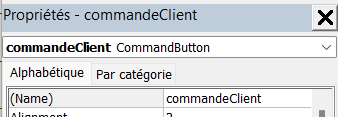
Dans le formulaire d’accueil « F\_Menu », nous allons ajouter un bouton qui indexe le formulaire client du nom « F\_client ».

Pour se faire :

📌Ajoutons un bouton « client » dans « F\_Menu » sans utiliser l’assistant.

📌Puis faire alt+f11 pour ouvrir le code.

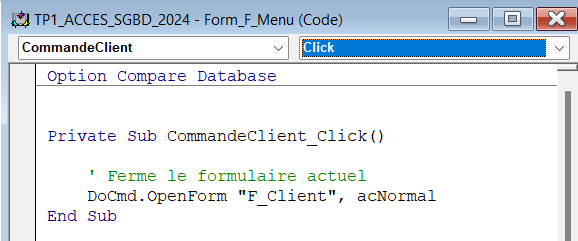
📌Renommer le nom de la variable « commandClient »



* + Saisissez ce code

### L’action qui declenche la commande

Il y a d’autres actions comme DbClick (double click) etc



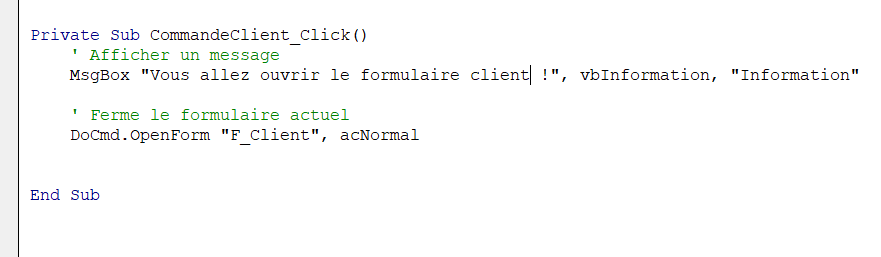
### Mode d’ouverture

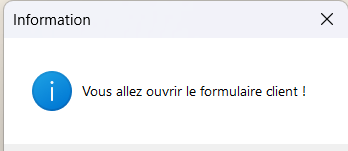
### Commentaire ; toute ligne commençant par ‘ est considérée comme commentaire

### Quelques paramètres du mode d’ouverture

* **acNormal** : mode Formulaire pour la saisie (ouverture par défaut)
* **acDesign** : mode Création
* **acLayout** : mode Page
* **acPreview** : mode Aperçu avant impression
* **acFormDS** : mode Feuille de données
  + **Ajoutons un message à l’ouverture du formulaire client**

**Le code devient**





Commentaire du code :

### MsgBox : une fonction permettant d’afficher un message

**vbInformation** : Affiche une icône d'information (un "i").

**"Information"** : Titre de la boîte de message

## Fermer le formulaire client

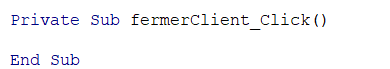
-Comme vu précédemment, ajouter un bouton fermer dans « F\_Client ».

-Dans l’editeur de code, renommer la commande du buton en « fermerClient »

- dans la liste des commandes, choisir « fermerClient »

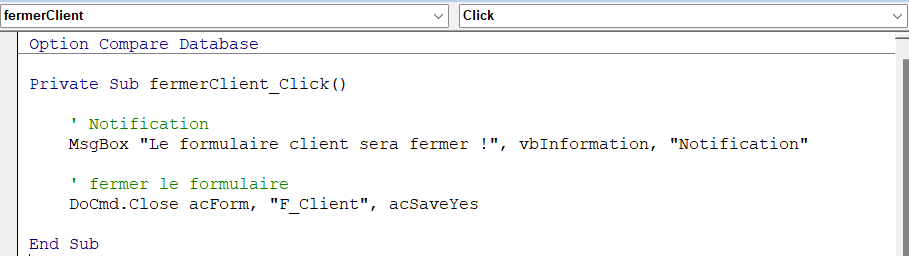


- vous aurez automatiquement



-Saisir le code entre private sub et End Sub

-prendre l’action « Click »



Action lors de la fermeture :

### acSavesYes: Sauvegarder les modifications

acSaveNo : ne pas sauvegarder

etc

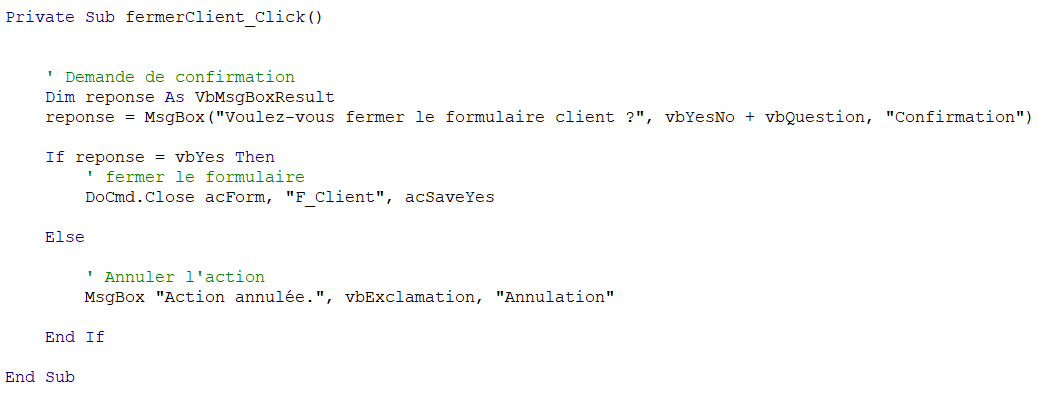
Type de l’objet:

### acForm: formulaire

actable : une table

etc

* + Demandons l’avis de l’utilisateur avant de fermer



Commentaires :

Dim : est le mot clé pour declarer une variable dans VBA ACCES

Syntaxe : Dim « nom de la variable » AS « type de la valeur à stockée »

Le parametre : vbYesNo permet de poser une question (oui/non) à l’utilisateur et la fonction MsgBox retourne vbYes ou vbNo.

vbQuestion : l’icone d’interrogation

**Icônes disponibles :**

**vbInformation** : Icône d'information.

**vbExclamation** : Icône d'avertissement.

**vbCritical** : Icône d'erreur.

**vbQuestion** : Icône de question.

**Boutons disponibles :**

**vbOKOnly** : Bouton "OK" uniquement.

**vbYesNo** : Boutons "Oui" et "Non".

**vbOKCancel** : Boutons "OK" et "Annuler".

Ajoutons un bouton d’enregistrement dans le formulaire client

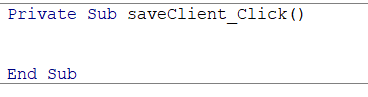
-nommer ce bouton « save »

-nommer la commande de ce bouton « saveClient »

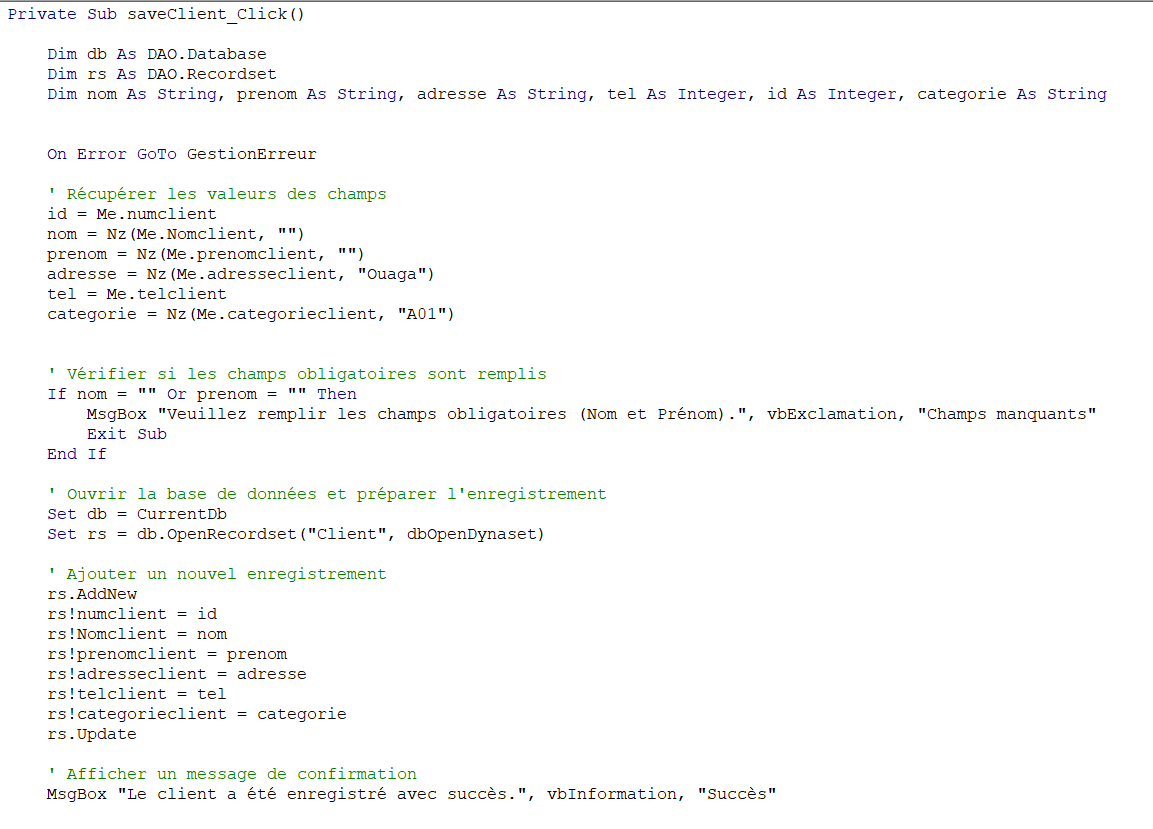
- dans la liste des commandes, choisir « saveClient »

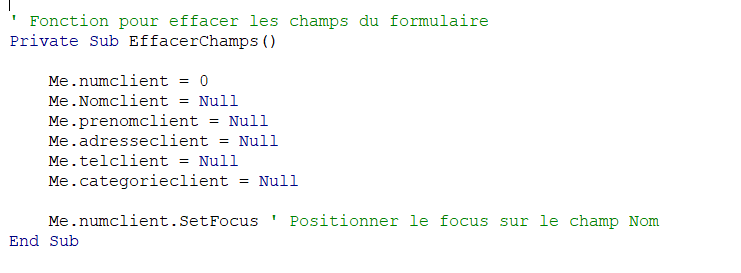


- vous aurez automatiquement



-Saisir le code entre private sub et End Sub





Commentaires :

Récupération des données des champs :

Nz(Me.NomClient, "") : Assure que même si le champ NomClient est vide, il retourne une chaîne vide ("") au lieu de Null.

Vérification des champs obligatoires :

Vérifie si les champs " NomClient " et "prenomClient" sont remplis avant d'enregistrer le client.

Si ces champs sont vides, le processus s'arrête et un message d'alerte est affiché.

Enregistrement dans la table :

db.OpenRecordset("Client", dbOpenDynaset) : Ouvre la table "Client".

rs.AddNew : Prépare l'ajout d'un nouvel enregistrement.

rs!nom, rs!prenom, etc. : Remplit les colonnes de la table avec les données récupérées du formulaire.

rs.Update : Enregistre les modifications dans la table.

Message de confirmation :

Affiche un message après un enregistrement réussi.

Effacement des champs :

Appelle la procédure EffacerChamps pour réinitialiser les champs du formulaire.

Gestion des erreurs :

Capture les erreurs potentielles et affiche une alerte avec Err.Description.