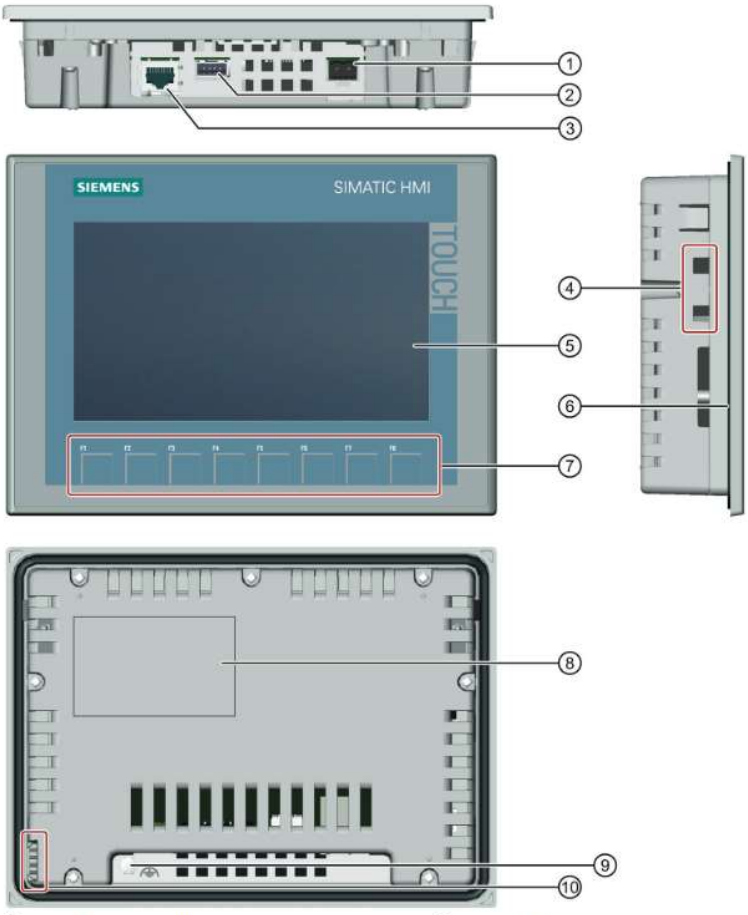
Dossier resource IHM SIEMENS Logiciel TIA PORTAL WinCC



① Connecteur d'alimentation
② Interface USB
③ Interface PROFINET
④ Encoches pour un clip de montage
⑤ Afficheur/écran tactile
⑥ Joint de montage
⑦ Touches de fonction
⑧ Plaque signalétique
⑨ Prise de terre fonctionnelle
⑩ Glissière des bandes de repérage

SOMMAIRE

Introduction

La plateforme **T**otally **I**ntegrated **A**utomation **Portal** est l'environnement de travail Siemens permettant de mettre en œuvre des solutions d'automatisation avec un système d'ingénierie intégré comprenant les logiciels :

- **SIMATIC STEP 7** permettant de programmer les automates programmables des gammes :
  - ✓ **S7-1200**
  - ✓ **S7-1500**
  - ✓ **S7-300**
  - ✓ **S7-400**
- **SIMATIC WinCC** permettant de programmer les **I**nterfaces **H**omme **M**achine

Généralités

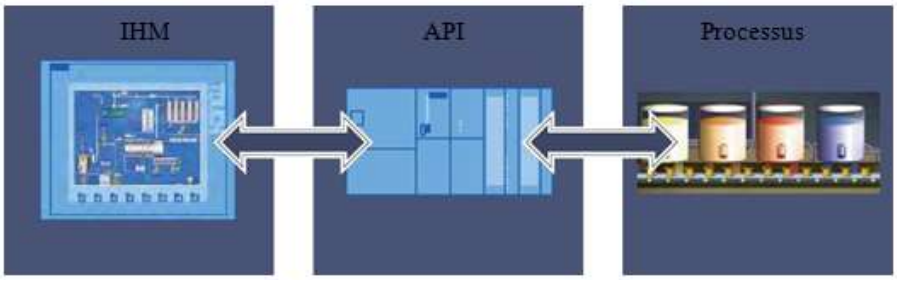
Le **dialogue homme/machine** est la fonction privilégiée par laquelle un opérateur peut à la fois surveiller et conduire un système automatisé.

La **qualité de l'interface** se mesure à la facilité avec laquelle le technicien d'exploitation ou de maintenance peut percevoir, comprendre et réagir efficacement à un évènement, et cela quelles que soient les conditions d'environnement. La **compréhension des informations** se fonde sur la bonne lisibilité et la précision de l'information. La **qualité** de la **réaction** doit être facilitée par une ergonomie soignée des organes de commande, tels que par exemple les surfaces des touches, leur accessibilité...

Les **terminaux de dialogue**, maillons essentiels de l'interface entre les automates programmables et leurs parties opératives, permettent :

- ❑ Un contrôle visuel ou une surveillance de l'état du procédé par des messages alphanumériques préprogrammés,
- ❑ Une signalisation et un enregistrement des défauts grâce à une table d'alarmes,
- ❑ Une impression horodatée des dysfonctionnements,
- ❑ Une commande de process par touches de fonction TOR.

Un système IHM constitue l'interface entre l'utilisateur et le processus. Le processus est pour l'essentiel piloté par l'automate. L'utilisateur peut visualiser le processus ou intervenir dans le processus en cours par le biais d'un pupitre opérateur.



Les possibilités suivantes vous sont offertes pour le contrôle-commande de machines et d'installations

- ❑ Représenter les processus
- ❑ Commander les processus
- ❑ Emettre des alarmes
- ❑ Gérer les paramètres du processus et les recettes

Création d'une table des variables API

❑ Introduction

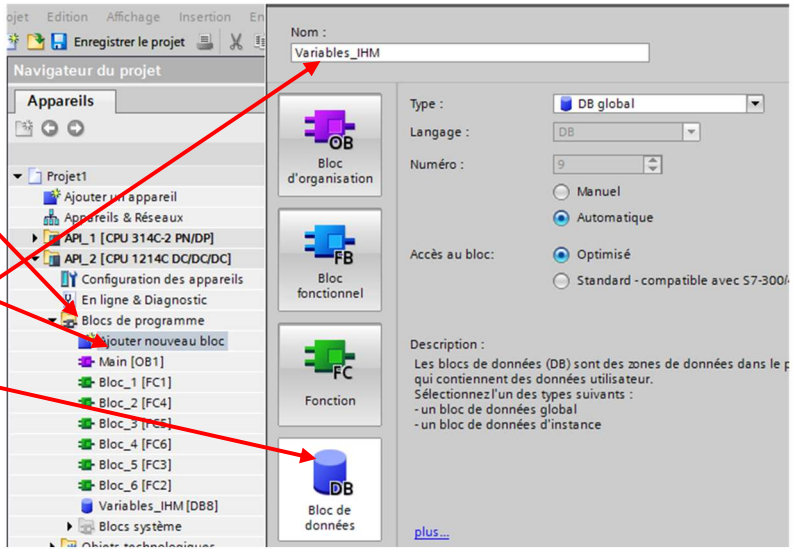
L'ensemble des variables utilisées dans un pupitre opérateur (boutons, voyants, variables numériques, …) sont adressées dans une table de variables API de l'automate connecté au pupitre opérateur.

Cette table de variables est réalisée **dans un bloc de données (DB)** dans lequel on définit les variables avec leur format. (Bool, Int, …)

Aucune adaptation dans l'IHM n'est donc nécessaire en cas de modification de l'adresse dans la table des variables API.

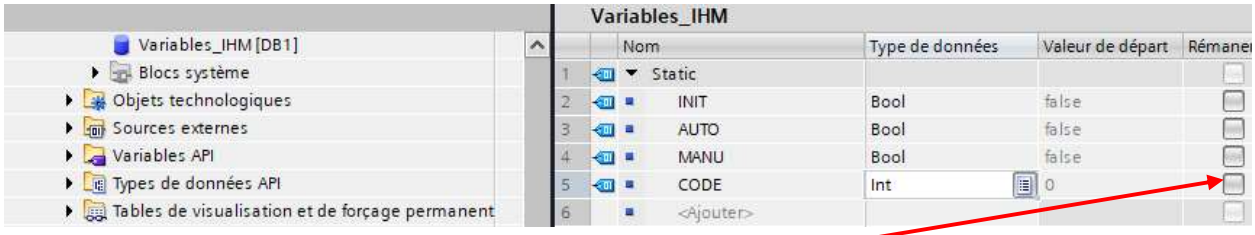
❑ Création d'un bloc de données

- Se positionner sur bloc de programme
- Cliquer sur ajouter nouveau bloc
- Choisir DB
- Indiquer le nom **Variables_IHM**



- Définir les variables IHM

On obtient une table comme ci-dessous



Remarque : La notion de rémanence permet de sauvegarder la valeur après coupure de courant.

Création d'un pupitre opérateur avec vue IHM

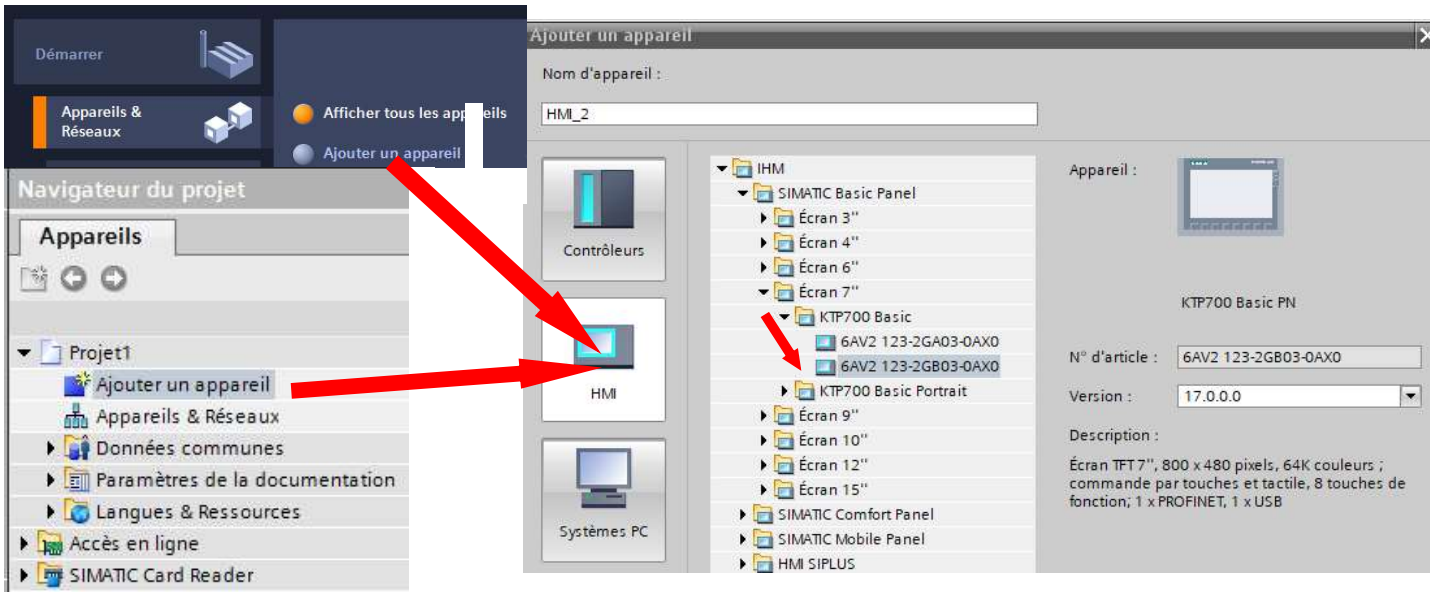
❑ Introduction

Les étapes suivantes décrivent comment créer un nouveau pupitre opérateur et un modèle pour la vue IHM.

Conditions requises

- Le programme a été créé.
- La vue du projet est ouverte.

❑ Ajout d'un nouveau pupitre opérateur

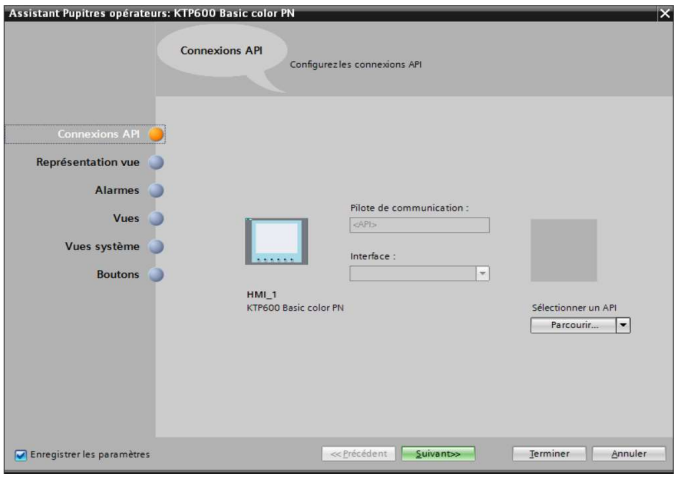


- Insérez un nouvel appareil par le biais du navigateur du projet ou vue du portail.
- Donnez-lui un nom et sélectionnez un pupitre opérateur. Ne désactivez pas la case d'option "Lancer l'assistant Appareils".

❑ Création d'un modèle pour une vue IHM

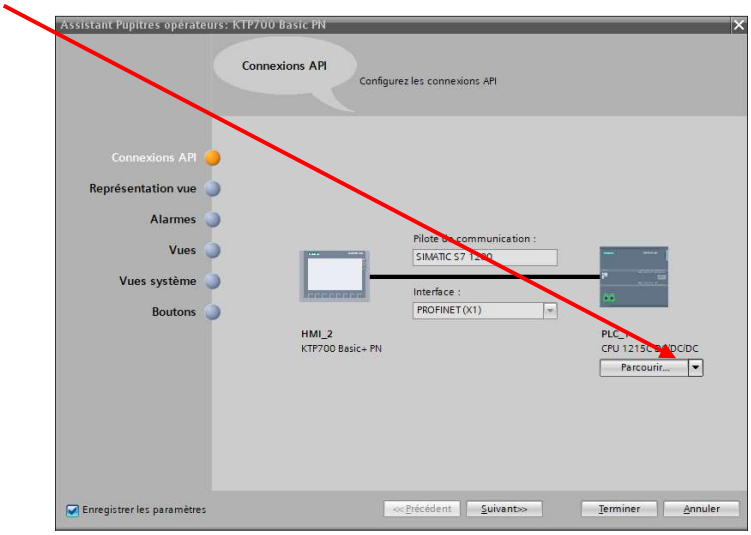
L'assistant des pupitres opérateur s'ouvre une fois que vous avez créé un pupitre opérateur. Cet assistant démarre en affichant une fenêtre dont le but est de renseigner les cinq éléments suivants :

- Connexion API.
- Représentation vue.
- Alarmes.
- Vues.
- Boutons.



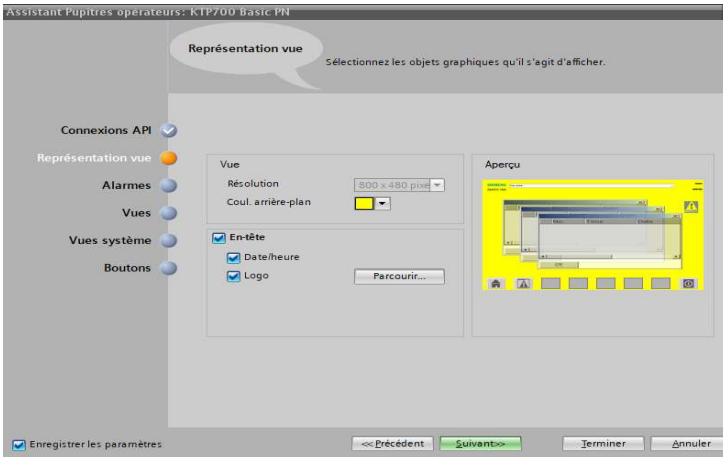
Connexion API

La configuration de la connexion entre le pupitre opérateur et l'automate s'effectuera à l'aide de



Représentation vue

Choisir la couleur d'arrière-plan pour le modèle et les éléments de la ligne d'en-tête.

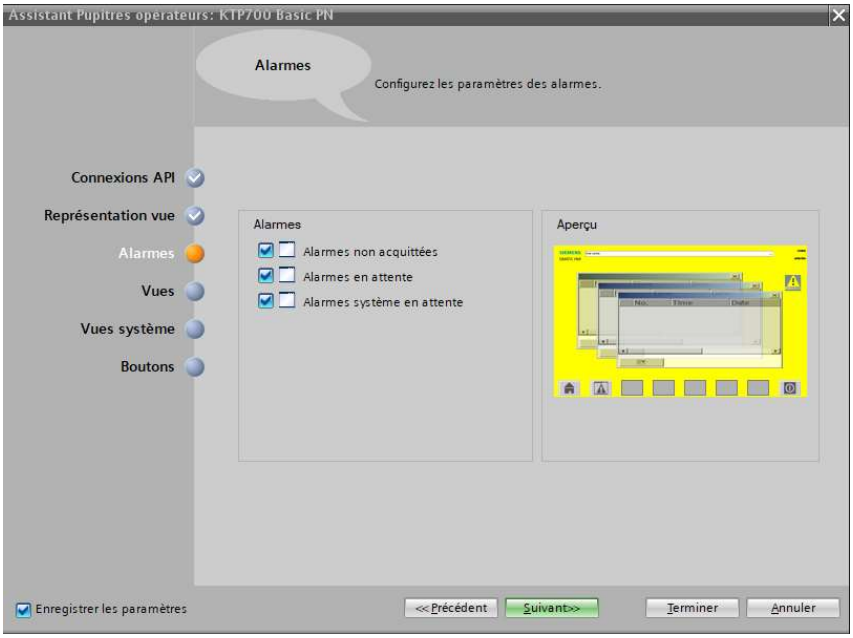


Remarque

Il sera possible de modifier les valeurs paramétrées ici pour la représentation de la vue à posteriori dans le modèle de la vue IHM.

Alarmes

Si dans le projet, il est nécessaire d'utiliser des alarmes, il faut les activer ici.



Remarques

Si vous activez les alarmes dans l'assistant des pupitres opérateur, vous pourrez sortir les messages par le biais du pupitre opérateur. Les fenêtres de message créées ici sont placées sous "Gestion des vues" dans la vue globale. Vous utiliserez des messages, par exemple pour émettre via le pupitre opérateur des avertissements en cas de dépassement d'une valeur limite. Il est possible de compléter les messages à volonté avec des informations supplémentaires, par exemple pour localiser plus facilement des perturbations dans le système. On distingue principalement entre messages utilisateur et messages système :

- Les messages utilisateur servent à surveiller le processus de l'installation.
- Les messages système sont importés dans le projet et contiennent des informations d'état du pupitre opérateur utilisé.

Vues

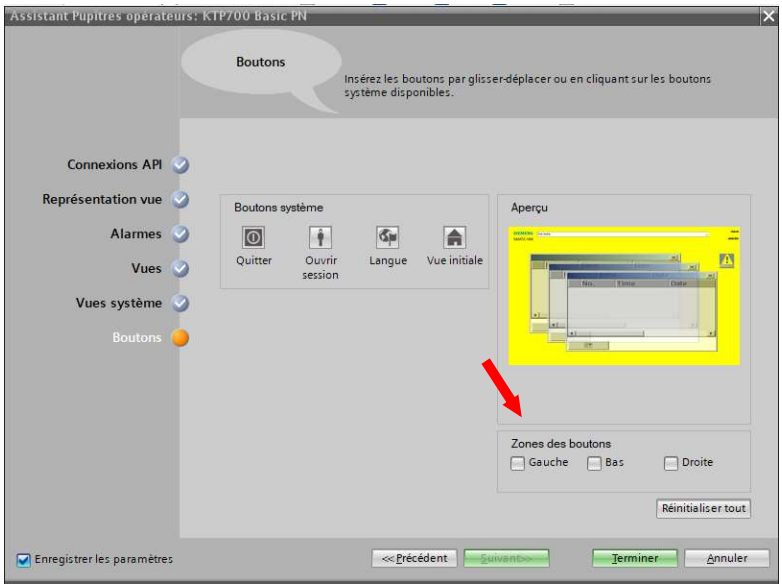
Ne pas paramétrer cet élément

Vues système

Désactiver les vues système

Boutons

Permet d'activer ou non la zone de boutons inférieure et d'insérer le bouton système "Quitter". Ce bouton système permet de quitter Runtime.



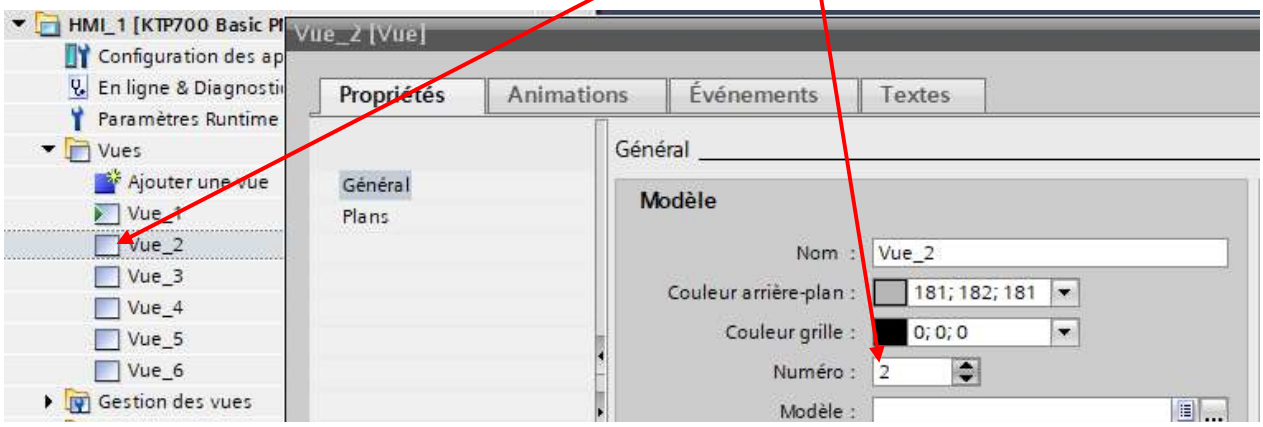
Après avoir cliqué sur « terminer » le logiciel a créé l'arborescence de l'IHM dans le navigateur de projet ainsi qu'une première vue appelée **vue racine** (c'est la page d'accueil de pupitre après mise sous tension). Après avoir ouvert cette vue, il est possible d'indiquer le message d'accueil de la machine.

❑ Création de nouvelles vues

Pour insérer de nouvelles vues, il suffit de cliquer sur ajouter une vue.

Lors de la création des vues **IHM**, bien vérifier les numéros de vue

*(on peut vérifier en cliquant **Clic droit → propriétés**)*



Remarque

Si la vue racine est affectée du numéro 1, modifier celui-ci avec un numéro non utilisé dans le projet.

❑ Création d'objets graphiques

Introduction

Dans le portail TIA, vous créez des vues pour le contrôle-commande de machines et d'installations. Pour créer des vues, vous disposez d'objets prédéfinis permettant de représenter votre installation, d'afficher des processus et de prédéfinir des valeurs de process. Les fonctions du pupitre opérateur déterminent la représentation du projet dans l'IHM ainsi que la fonctionnalité des objets graphiques.

Objets graphiques

On appelle objets graphiques tous les éléments que vous pouvez utiliser pour la représentation du projet dans l'IHM. Il s'agit, par exemple, de textes, boutons, schémas ou graphiques pour la représentation de parties de l'installation.

Utilisation d'objets graphiques

Il est possible de représenter les objets graphiques statiquement ou de les utiliser comme objets dynamiques à l'aide de variables :

- Les objets statiques ne changent pas en Runtime.
- Les objets dynamiques varient en fonction du processus. Vous visualisez les valeurs de process actuelles par le biais :
  - de variables API dans la mémoire de l'automate,
  - de variables internes dans la mémoire du pupitre opérateur, au moyen d'affichages alphanumériques, de courbes et de bargraphes.

  Font également partie des objets dynamiques les champs de saisie sur le pupitre opérateur qui permettent l'échange de valeurs de process et d'entrées de l'opérateur entre l'automate et le pupitre opérateur au moyen de variables.

Création et configuration d'un bouton "DCY"

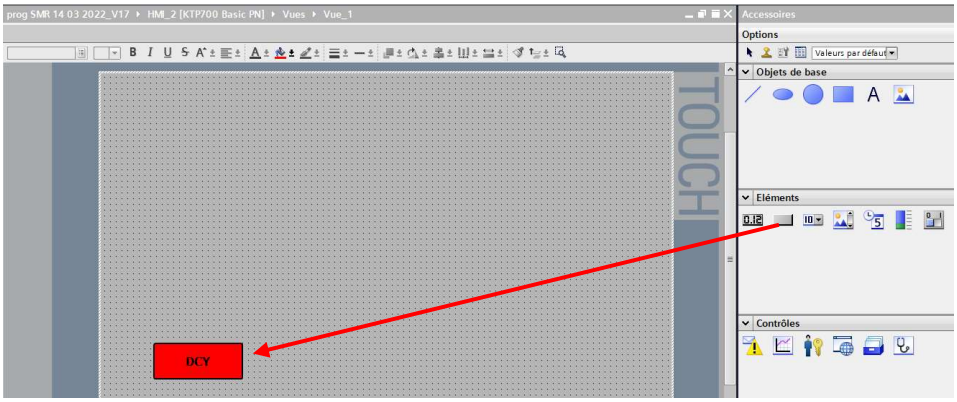
Les étapes suivantes décrivent comment créer le bouton "DCY" et le relier à une variable API par le biais d'une variable IHM externe.

Une variable IHM externe permet d'accéder à une adresse dans l'automate. Il est possible ainsi, par exemple, d'entrer une valeur de process par le biais d'un pupitre opérateur ou de modifier directement des valeurs de process du programme de commande par le biais d'un bouton.

L'adressage se fait au moyen de la table des variables API **(dans le bloc de données DB que l'on a créé précédemment)** de l'automate connecté au pupitre opérateur. La variable API est associée à la variable IHM par le biais du nom symbolique. Aucune adaptation dans IHM n'est donc nécessaire en cas de modification de l'adresse dans la table des variables API.

- Ouvrir une vue

Fenêtre de travail



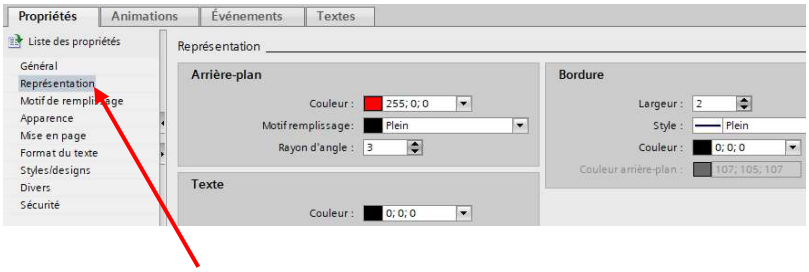
Task Cards :

Onglet de sélection des tâches

- Sélectionner un bouton dans la **fenêtre Task Cards** et par glisser & déposer le positionner dans le pupitre de la fenêtre de travail.

➢ Onglet Propriétés

Dans cet onglet, renseigner les paramètres généraux : format police, couleur bouton, …….



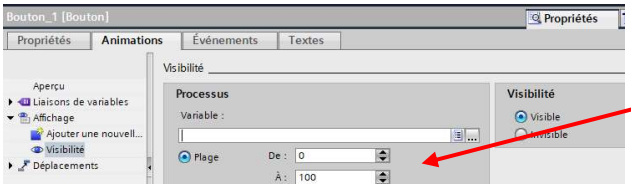
Fenêtre d'inspection

- Dans la **fenêtre d'inspection**, utiliser les différents points de l'onglet Attributs pour renseigner ce bouton (couleur, taille, …)

Remarque
De la même manière, il est possible d'insérer des de champs de variable, des images, des formes, du texte, des bargraphes, du texte, des courbes, …….

➢ Onglet Animation

Dans cet onglet, on renseigne sur la **visibilité d'un bouton** par exemple



On clique sur le bouton et on indique que celui-ci sera visible en fonction de la valeur d'une variable.

Il est également possible d'effectuer des déplacements d'objets (simuler par exemple le déplacement d'un vérin)

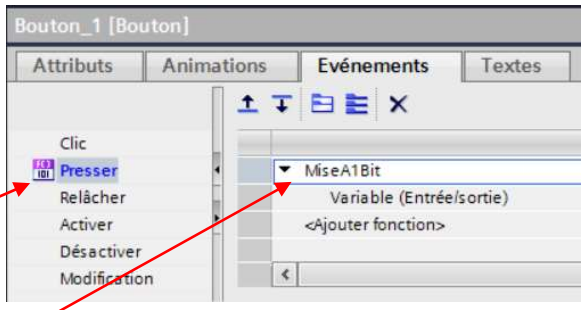
➢ Onglet Evènements

Dans cet onglet, on renseigne sur les différentes fonctions de l'élément

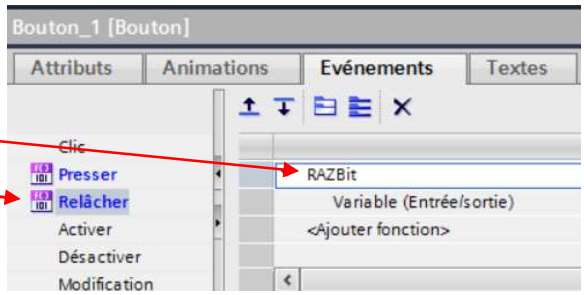
- Dans la fenêtre d'inspection, ouvrir **l'onglet Evénements** puis choisir l'action qui doit être réalisée après appui sur le bouton.

Par exemple pour un bouton poussoir,

au "**Presser**", on met à 1 la valeur du bit

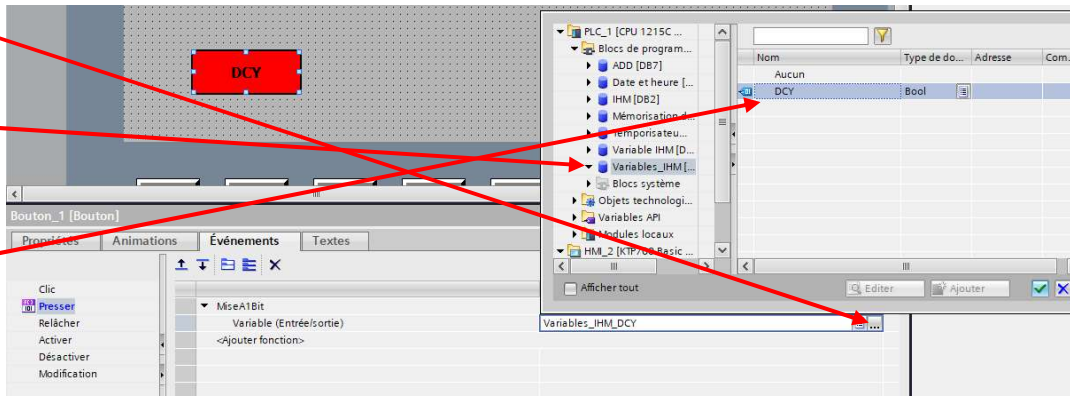


au "**Relâcher**", on met à zéro la valeur du bit.



- Après validation, indiquer l'adresse de la variable API

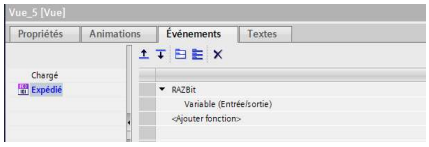
- Cliquer sur
- Ouvrir le bloc Variables_IHM
- Choisir la variable associée au bouton



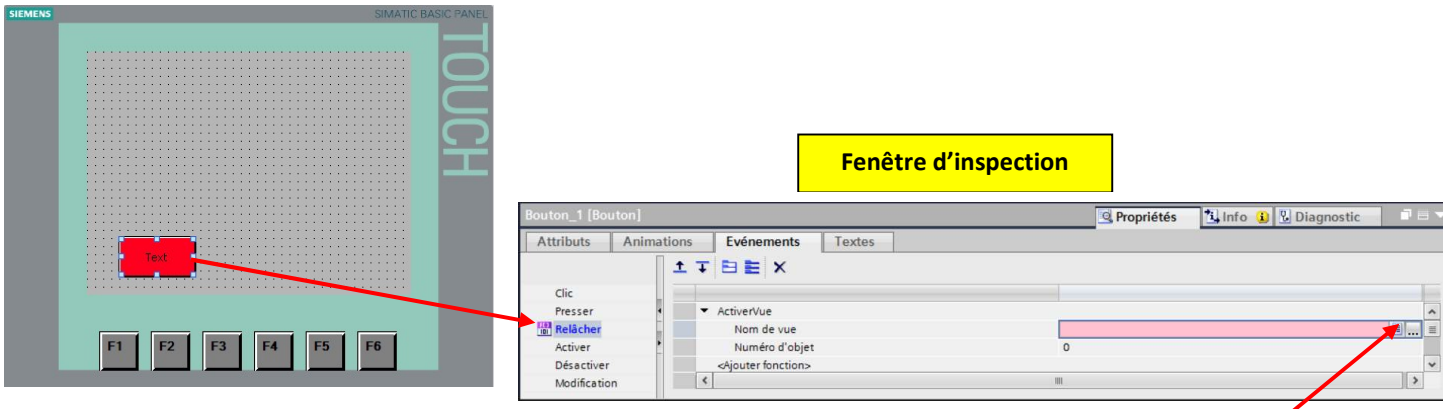
Remarque
Il est fortement de recommandé de remettre à zéro la valeur du bit lorsque l'on quitte la vue.

- Se positionner sur la vue puis dans l'onglet Evènement :

Cliquer sur Expédié puis remettre à zéro le ou les bits concernés

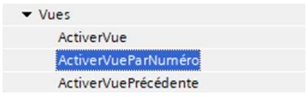


❑ Appel de vue par action sur un objet graphique



- Cliquer sur l'objet graphique (ici **bouton**)
- Dans la fenêtre d'inspection, cliquer sur **Evénements**
- Choisir Evénement au **Relâcher**
- Choisir **vues** dans la liste puis **ActiverVue** et choisir la vue désirée

Remarque : Il est possible d'activer la vue par son nom, par son numéro ou la vue précédente.



❑ Appel de vue depuis une étape d'un grafcet

➢ Principe général

Les pointeurs de zone sont des tableaux de paramètres. Ces tableaux de paramètres fournissent à WinCC en Runtime des informations sur le temps d'exécution du projet. Ces informations concernent la position et la taille des zones de données dans l'automate.

Au cours de la communication, l'automate et le pupitre opérateur écrivent et lisent tour à tour des données dans ces zones de données.

L'évaluation des données enregistrées dans ces plages de données permet à l'automate et au pupitre opérateur de déclencher des actions prédéfinies.

Fonction

La boîte de tâches API permet de fournir des tâches API au pupitre opérateur et ainsi de déclencher des actions sur ce dernier. Parmi ces fonctions, on distingue p. ex. :

- Afficher la vue
- Réglage de la date et de l'heure.

Structure des données

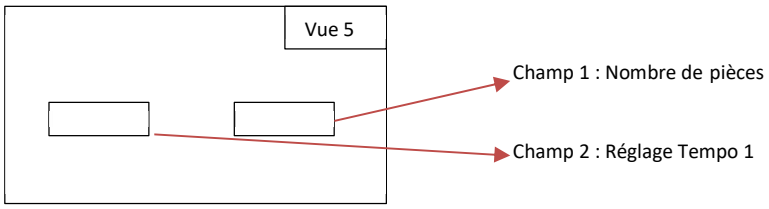
Le numéro de tâche figure dans le premier mot de la boîte de tâches API. Suivant la tâche API concernée, jusqu'à trois paramètres peuvent être transférés.

Mot	Octet de poids fort	Octet de poids faible
n + 0	0	Numéro de la tâche
n +1	Paramètre 1	
n + 2	Paramètre 2	
n + 3	Paramètre 3	

Pour appeler une vue, le numéro de la tâche est 51

Mot	Octet de poids fort	Octet de poids faible
n + 0	0	51
n +1	Numéro de vue	
n + 2	Non utilisé	
n + 3	Numéro de champ	

Remarque : Si on renseigne le numéro du champ (par exemple 1), on ouvre directement le champ correspondant au nombre de pièces avec accès au clavier pour définir le nombre désiré.

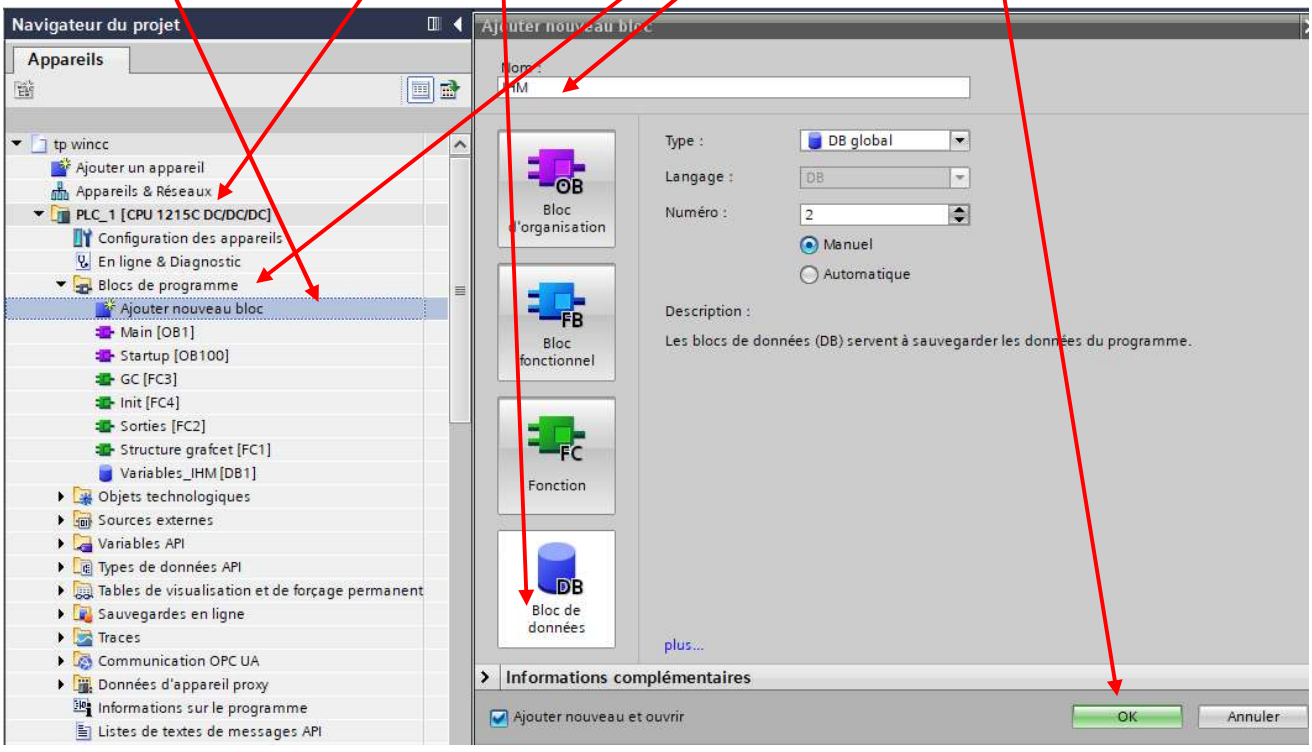


Il faut donc déclarer une zone de 4 mots

➢ Programmation API (déclaration d'une zone de 4 mots dans un bloc de données)

Création d'un bloc de données

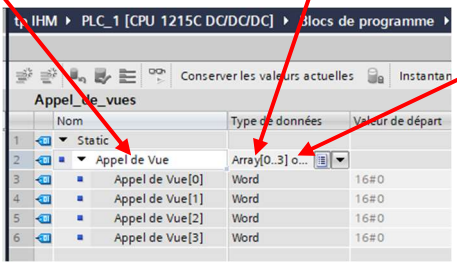
Dans le navigateur, se positionner sur API puis bloc de programme puis ajouter un nouveau bloc. Sélectionner bloc de données (DB) puis donner un nom (IHM) puis valider.



Déclaration de la zone de 4 mots (variables)

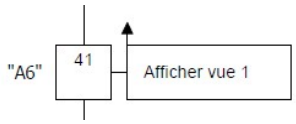
Créer un tableau de 4 mots

Dans le DB, donner le nom puis indiquer le type « ARRAY » puis modifier les valeurs 0…3, type par WORD. Ceci créer un tableau de mots d'une longueur de 4.

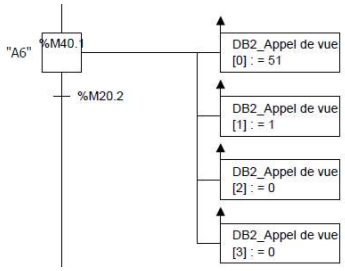


➢ Ecriture grafcet d'un appel de vue

Extrait Grafcet point de vue PC



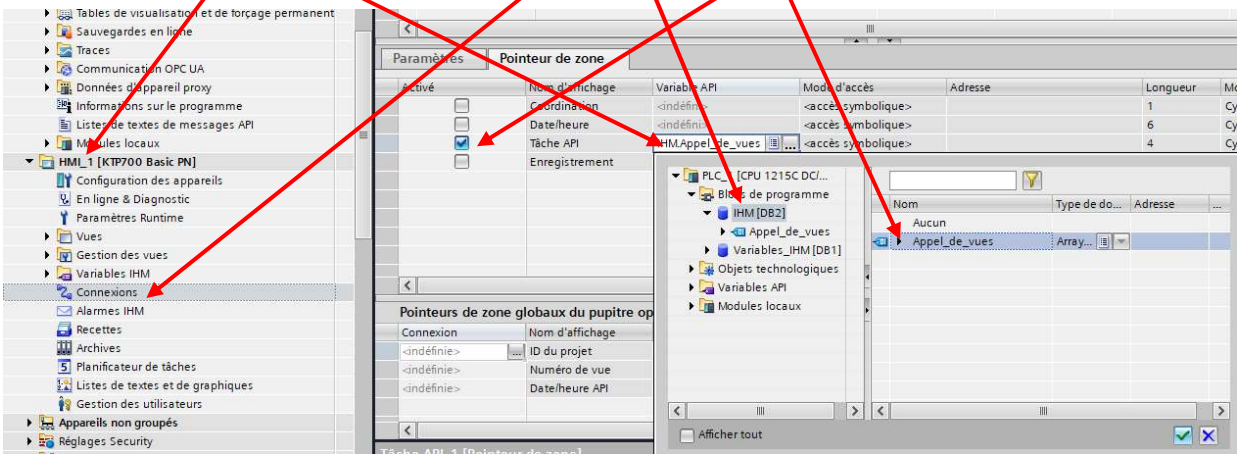
Extrait grafcet point de vue API



➢ Programmation IHM (paramétrage Connexions)

Avant d'utiliser un pointeur de zone, activez les pointeurs de zone sous "Connexions > Pointeur de zone". Paramétrez ensuite les pointeurs de zone.

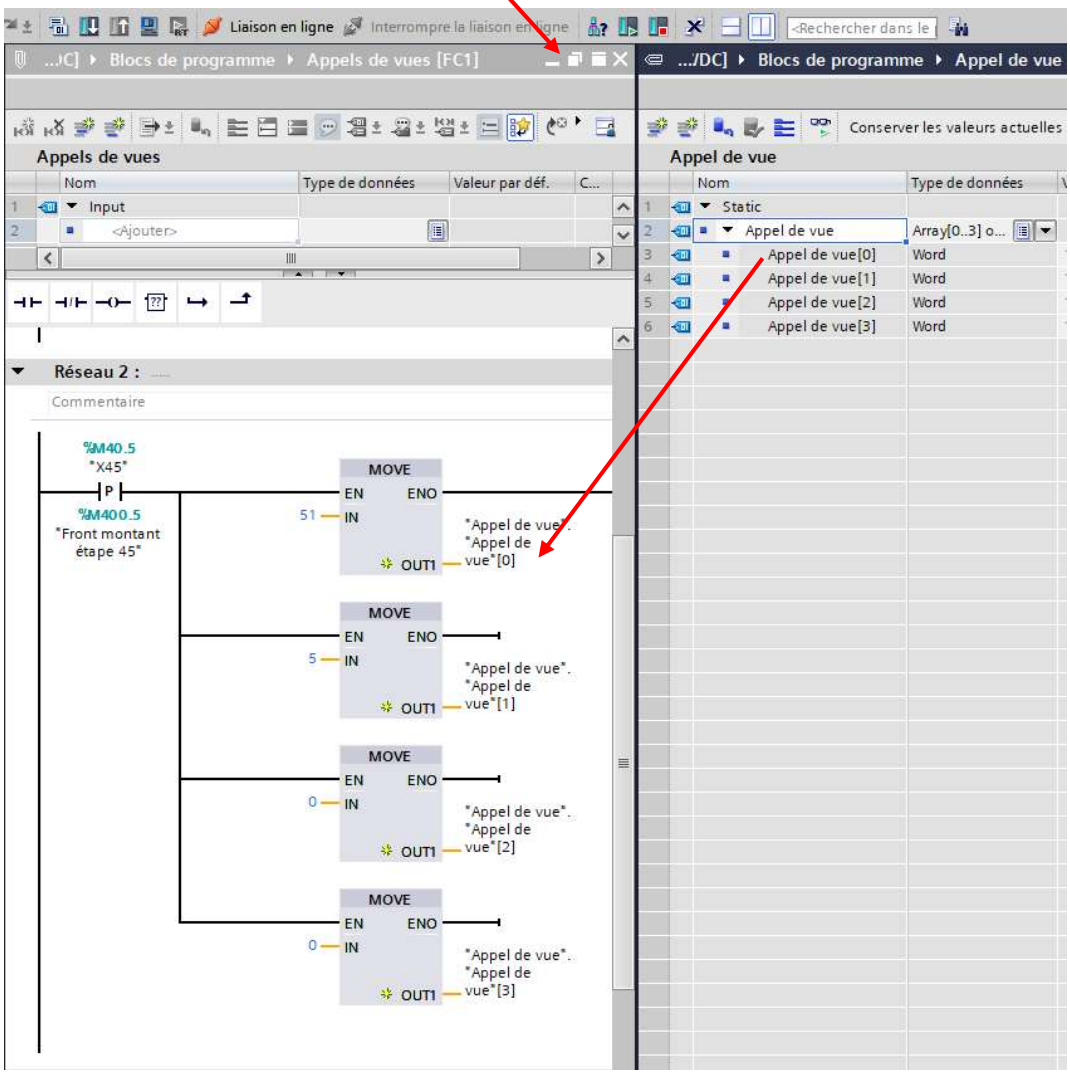
Dans le navigateur, se positionner sur HMI puis Connexions puis pointeur de zone puis cocher tâche API puis se positionner sur variable API. Choisir le numéro du DB puis le nom.



➢ Programmation de l'appel des différentes vues dans le programme API

- Ouvrir le bloc FC appelant les différentes pages et le bloc DB crée précédemment puis cliquer sur l'icône pour avoir les 2 fenêtres simultanément.
- Insérer les 4 blocs MOVE et par **glisser & déposer** indiquer les variables de sortie de ces blocs.
- Il est préférable d'appeler les vues de l'IHM sur front montant d'étape. Dans le programme ci-dessous, on appelle la vue 5 sur front montant de l'étape 45.

Mot	Octet de poids fort	Octet de poids faible
n + 0	0	51
n +1	5	
n + 2	0	
n + 3	0	



Remarque :

La configuration, au départ, paraît longue, mais ensuite il suffit de faire un copier /coller du réseau et de modifier uniquement l'étape et la vue appelée. S'il est nécessaire d'appeler un champ précis dans une vue, il suffit de l'indiquer dans le troisième mot.

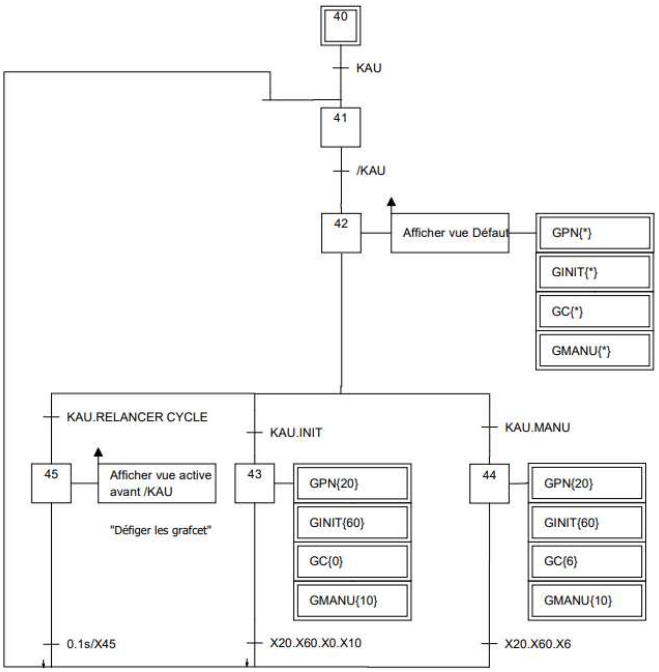
❑ Mémorisation numéro de vue IHM

➢ Problématique

Dans certains cas, il est utile de mémoriser la dernière vue affichée par l'IHM. C'est le cas après un figeage dans le grafcet de sécurité ; si on désire reprendre le cycle automatique après cette action, il faut défiger et afficher la vue qui était active avant le figeage.

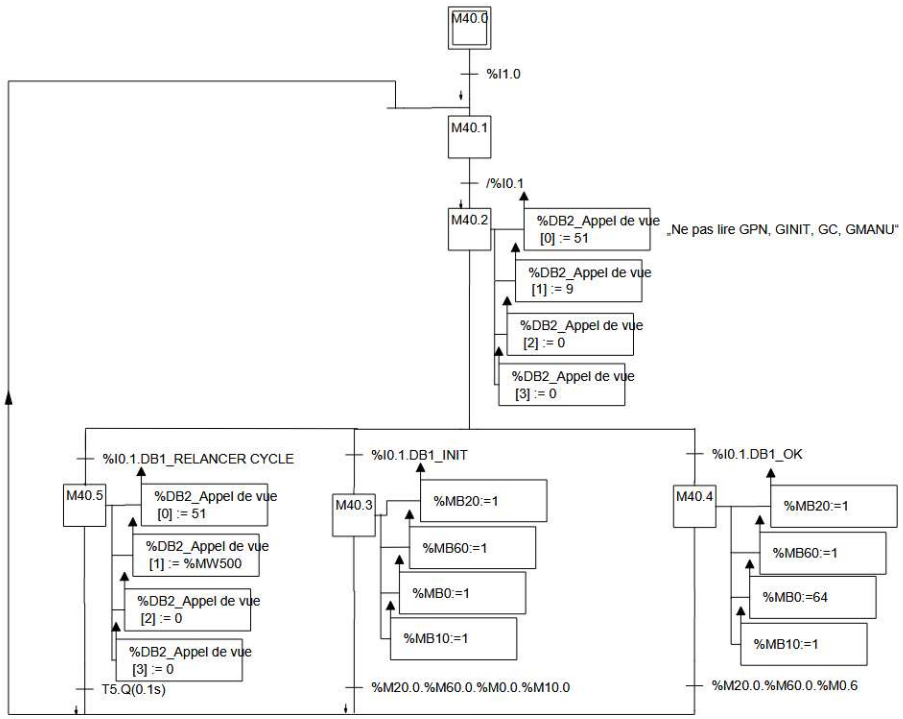
➢ Mise en situation

Grafcet de sécurité point de vue PC



Grafcet de sécurité point de vue API

%MW500 :
stockage mémorisation vue



➢ Pointeur de zone « numéro de vue »

Dans ce pointeur de zone, les pupitres opérateur déposent des informations concernant la vue appelée sur le pupitre opérateur concerné.

Il est ainsi possible de transférer des informations sur le contenu actuel de la vue à l'automate

Le pointeur de zone est une zone de données d'une longueur fixe de 5 mots dans la mémoire de l'automate.

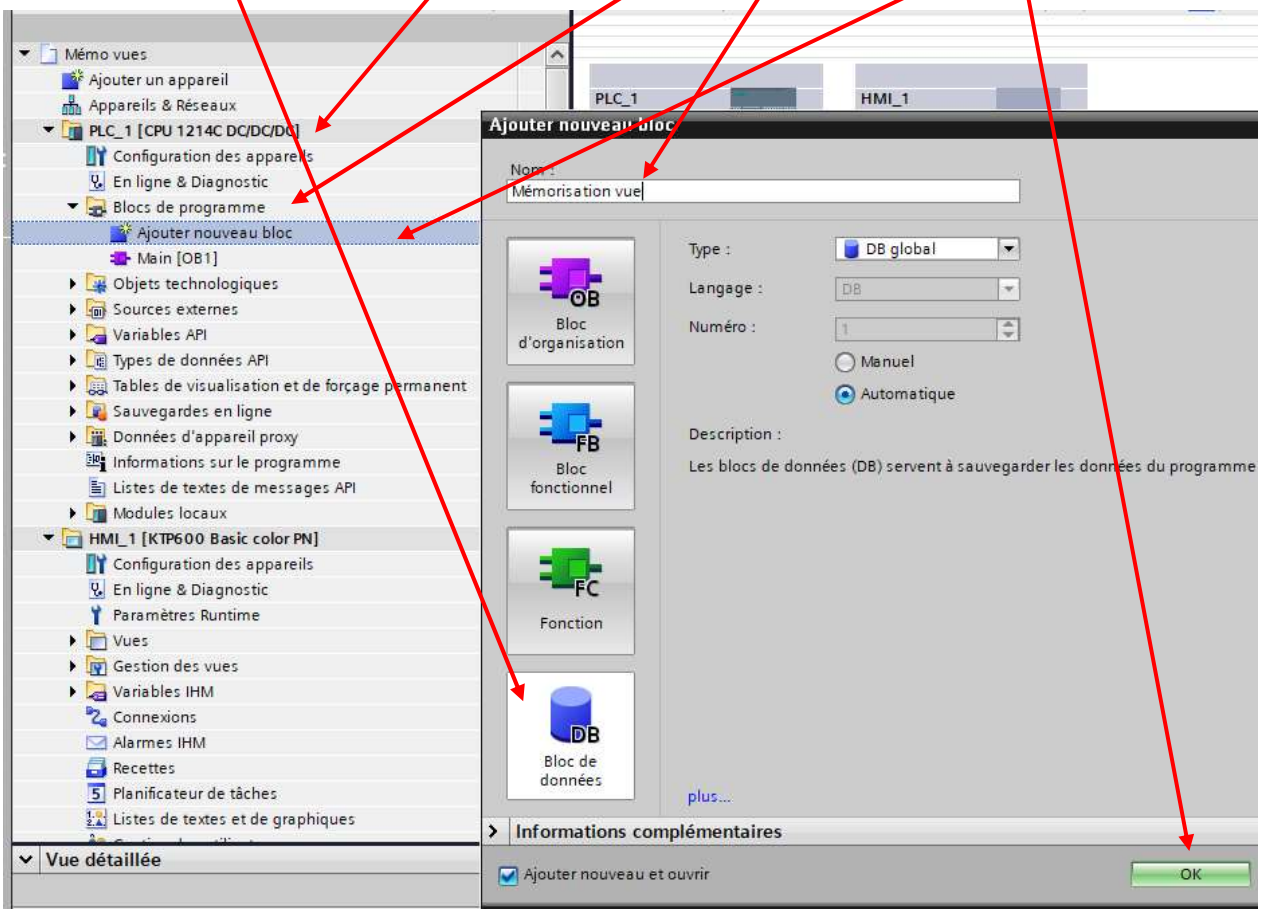
Type de vue actuel
N° de vue actuel
Réservé
N° de champ actuel
Réservé

➢ Programmation API (création bloc de données)

Déclaration de la zone de 5 mots dans un bloc de données (DB).

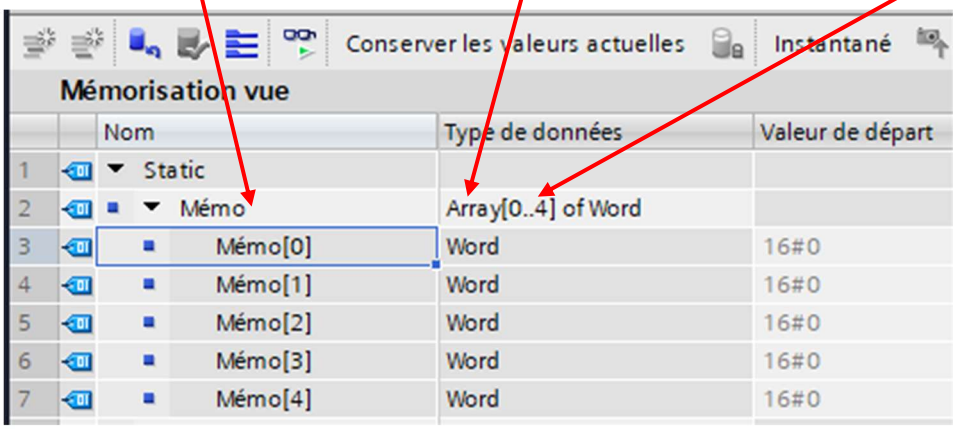
- **Création d'un bloc de données**

Dans le navigateur, se positionner sur API puis bloc de programme puis ajouter un nouveau bloc. Sélectionner bloc de données (DB) puis donner un nom (Mémorisation vue) puis valider.



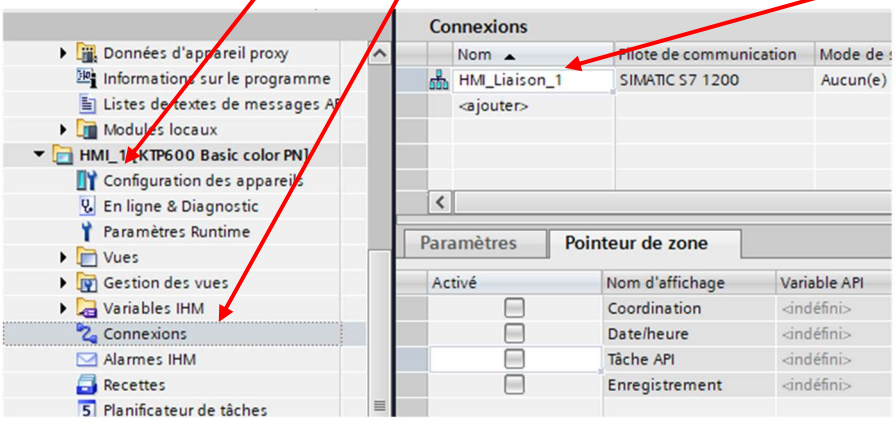
Déclaration de la zone de 5 mots (variables)

Dans le DB, donner le nom puis indiquer le type « ARRAY » puis modifier les éléments (remplacer 1 par 4, type par WORD. Ceci crée un tableau de mots d'une longueur de 5.

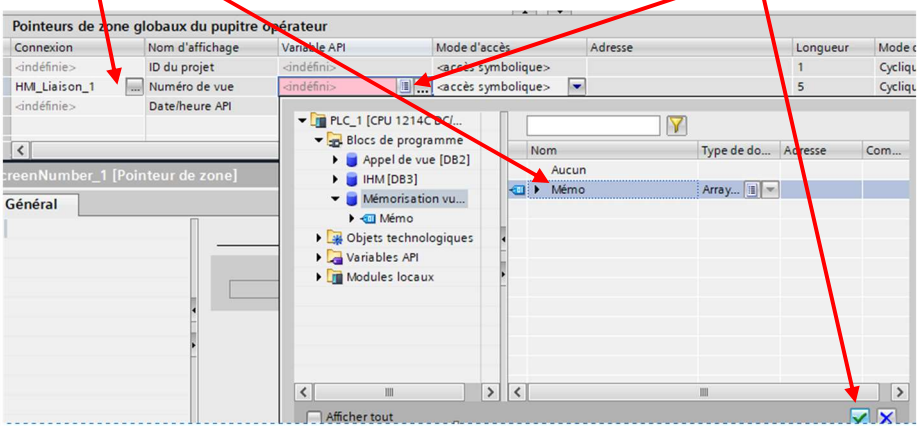


➢ Programmation IHM (paramétrage Connexions)

Dans le navigateur, se positionner sur IHM puis connexions. Observer le nom de la liaison API-IHM.



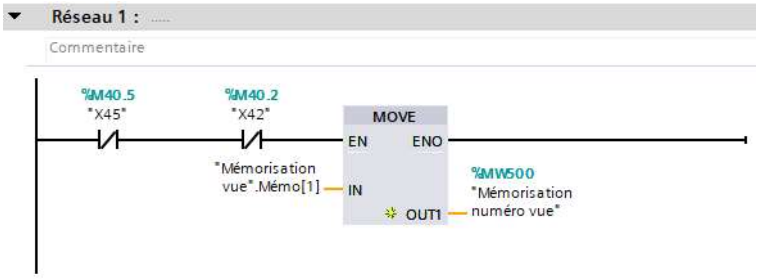
Dans le tableau **pointeurs de zone globaux du pupitre opérateur**, se positionner sur la ligne « Numéro de vue » puis choisir la liaison (identique à l'écran précédent) puis se positionner sur variable API, choisir le tableau (ARRAY) Mémo du DB Mémorisation de vue crée précédemment puis valider.



➢ Programmation API (Mémorisation de la dernière vue)

Le but est de stocker en permanence sauf à l'étape X45 et à l'étape X42 la valeur de la vue active dans le mot %MW500.

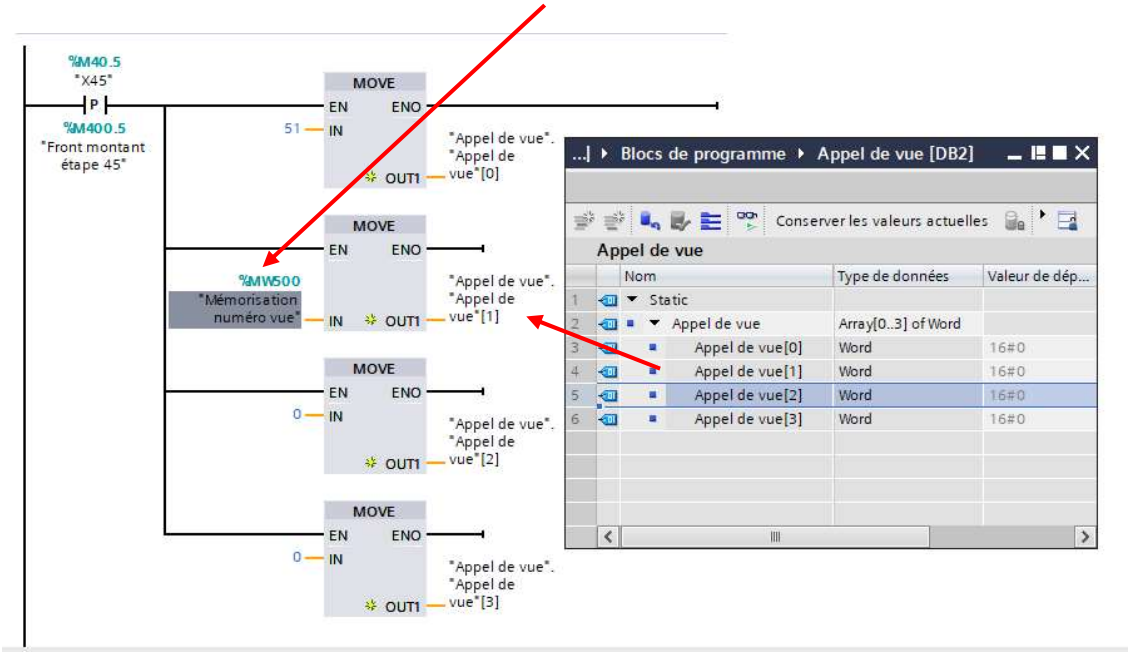
- Ouvrir le bloc FC appelant les différentes vues et le bloc DB mémorisation vue puis cliquer sur l'icône pour avoir les 2 fenêtres simultanément.
- Insérer le réseau ci-dessous dans le premier réseau. Pour insérer la variable *IN du bloc MOVE*, utiliser la fonction glisser & déposer ; cette variable est le mot 1 du DB contenant la valeur du numéro de vue.



➢ Programmation API (Programmation appel de vue de l'étape X45)

- Ouvrir le bloc FC appelant les différentes vues et le bloc DB Appel de vue puis cliquer sur l'icône pour avoir les 2 fenêtres simultanément.
- Insérer les 4 blocs MOVE et par glisser & déposer indiquer les variables de sortie de ces blocs

Il est préférable d'appeler les vues de l'IHM sur front montant d'étape. Dans le programme ci-dessous, on appelle la vue contenue dans le mot %MW500 sur front montant de l'étape 45.



Synchronisation de l'heure

➢ Problématique

Pour que toute l'installation affiche la même heure, vous pouvez synchroniser l'heure des différents composants de l'installation à l'aide de la fonction de synchronisation de l'heure. L'option de synchronisation de l'heure de WinCC fonctionne sous forme de système maître/esclave.

Pour que tous les composants d'une installation fonctionnent à une heure identique, un composant de système doit être défini comme horloge de base pour tous les autres composants. Le composant faisant fonction d'horloge de base est désigné comme horloge-maître. Les composants synchronisés recevant l'heure sont les horloges-esclaves.

Attributs de la synchronisation de l'heure

- Le pupitre opérateur peut donner l'heure en tant que maître, ou reprendre l'heure de l'automate en tant qu'esclave.
- En "mode maître", l'horloge est synchronisée à chaque établissement de liaison.
- En "mode esclave", l'horloge est synchronisée à chaque établissement de liaison, puis toutes les 10 minutes.
- La première synchronisation de l'heure est réalisée directement après le démarrage du Runtime sur le pupitre opérateur.
- La synchronisation de l'heure n'est effectuée que pendant le fonctionnement du Runtime sur le pupitre opérateur.

Remarque

Dans la configuration, sélectionnez un cycle d'acquisition du pointeur de zone Date/heure API qui ne soit pas trop court, car ceci influe sur les performances du pupitre opérateur. Recommandation : Cycle d'acquisition d'une minute, si votre process le permet.

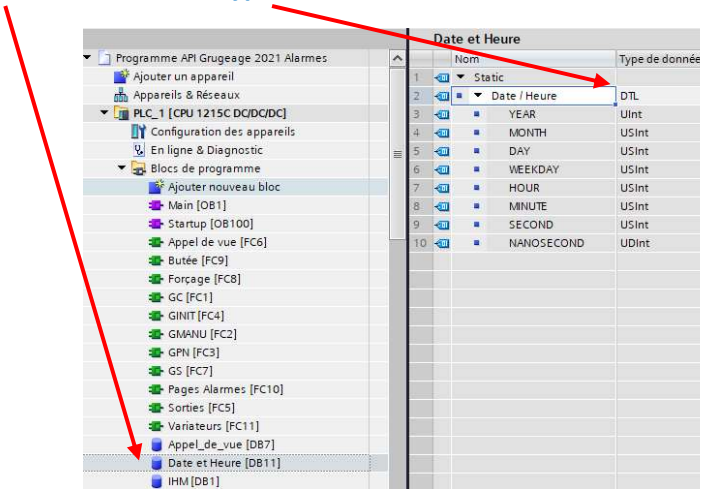
➢ Synchronisation de l'heure IHM en mode esclave et API en maître

"Date/Heure API" est un pointeur de zone global, il ne peut être configuré qu'une seule fois dans le projet.

Lorsque l'automate est configuré en tant que maître d'horloge, l'automate charge la zone de données du pointeur de zone. Le pupitre opérateur lit périodiquement les données par le biais du cycle d'acquisition configuré et se synchronise.

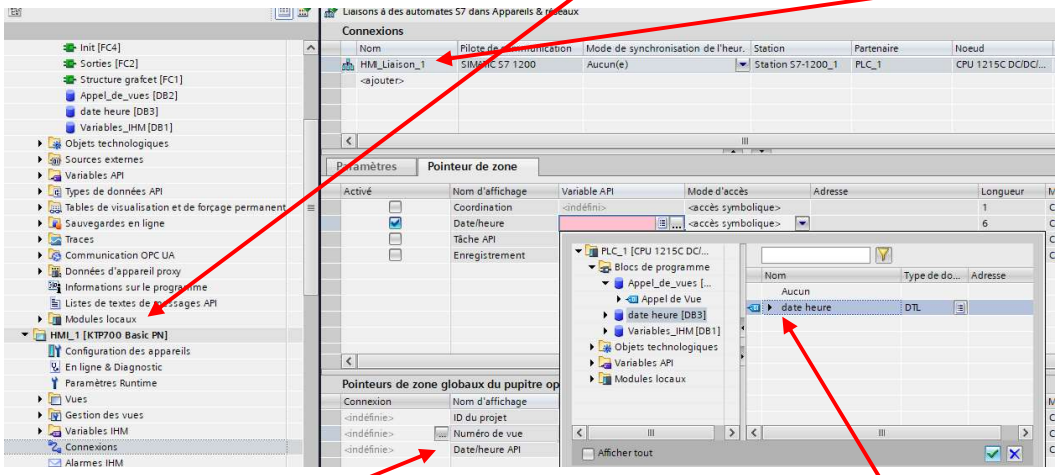
➢ Programmation API (création bloc de données)

Création d'un bloc de données Date et Heure de type DTL



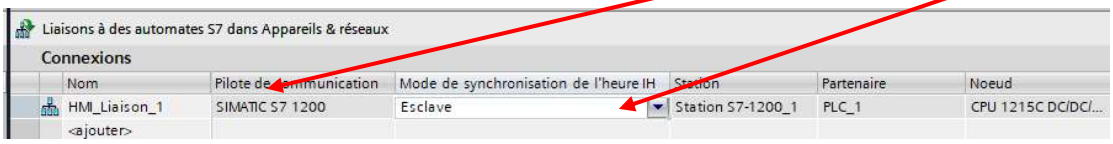
➢ Programmation IHM (paramétrage Connexions)

Dans le navigateur, se positionner sur IHM puis connexions. Observer le nom de la liaison API-IHM.



Dans le tableau **pointeurs de zone globaux du pupitre opérateur**, se positionner sur la ligne « Date/heure API » puis se positionner sur variable API, choisir le **DB Date et Heure** crée précédemment puis valider.

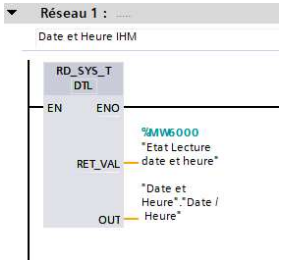
- Dans la colonne "Pilotes de communication", sélectionnez l'automate « **SIMATIC S7 1200** »
- Dans la colonne "Mode de synchronisation de l'heure IHM", sélectionnez « **Esclave »**



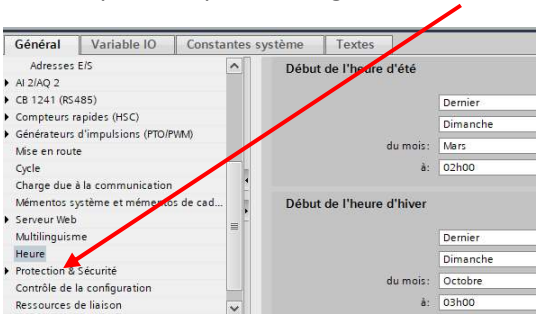
➢ Programmation API (Lecture Date et Heure API)

Le but est de lire l'heure et la date de la CPU et de la stocker dans le paramètre OUT DB Date et Heure crée précédemment. Le paramètre RET_VAL nous indique l'état de l'instruction.

- Ouvrir le bloc OB1 et insérer le réseau ci-contre



- Ne pas oublier de compléter le paramétrage de l'heure dans la **configuration de la CPU (en ligne)**

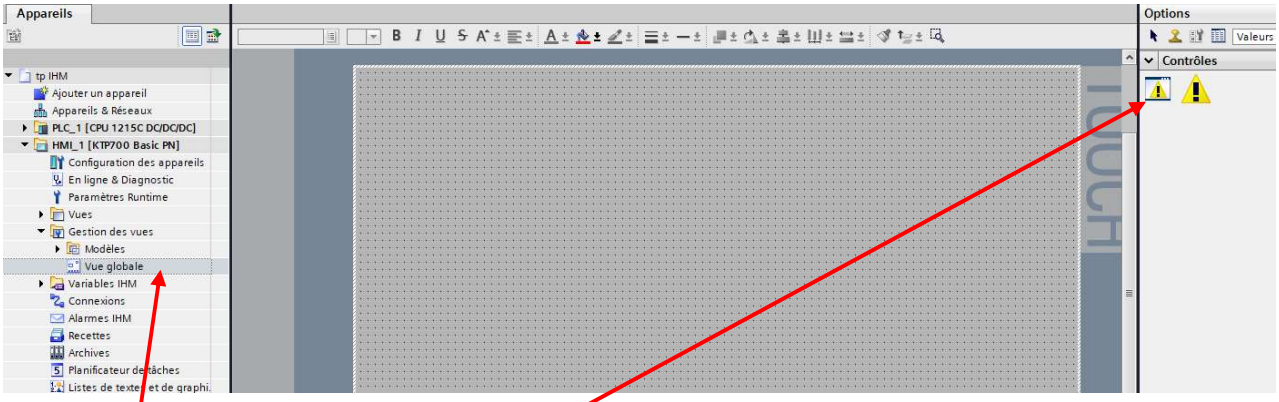


Alarmes de bit IHM

➢ Problématique

Les alarmes de bit IHM indique les changements d'état dans une installation et ils sont déclenchées par l'automate (exemple un défaut de pression ou un défaut thermique moteur). Ces alarmes apparaissent dans des fenêtres qui vont s'afficher **au premier plan** sur l'IHM.

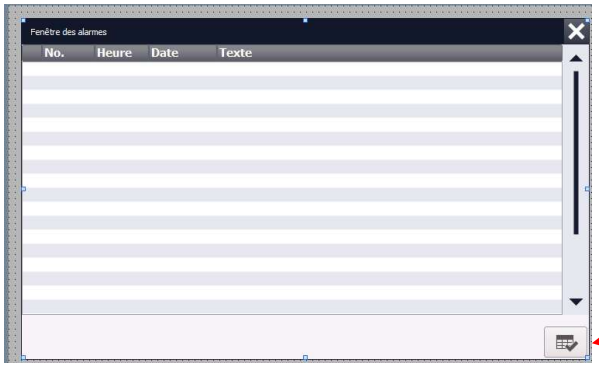
➢ Paramétrage IHM (paramétrage vue globale)



- Se positionner sur « Vue globale »
- Une fois dans « Vue globale », on insère la fenêtre des alarmes sur la vue globale à l'aide la fonction **glisser & déposer**

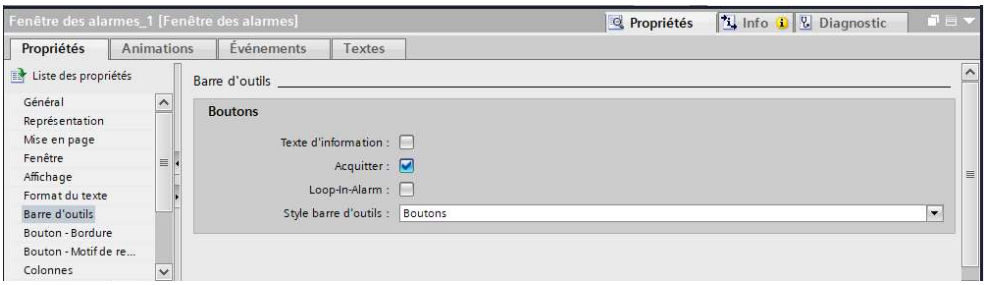
Remarque : il est possible que la fenêtre des alarmes soit déjà présente sur la vue globale.

On obtient cet écran



Bouton pour acquitter le défaut

Dans la fenêtre d'inspection, vous modifiez les paramètres de position, géométrie, style, couleur et police de l'objet, bouton acquittement ….



Eléments de commande

Les éléments servant à commander la vue des alarmes au runtime sont déterminés dans la fenêtre d'inspection, sous "Attributs > Affichage > Paramètres". Le tableau suivant présente les éléments de commande de la fenêtre des alarmes avec leur fonction :

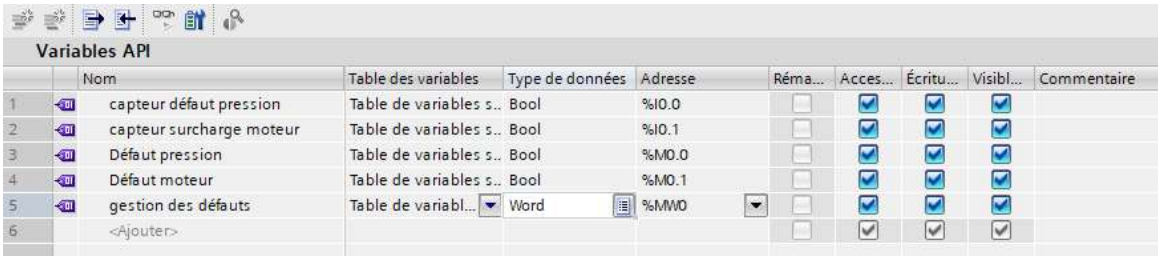
Bouton		Fonction
"Info-bulle"	?	Affiche l'info-bulle pour une alarme.
"Acquitter"	!	Acquitte une alarme.
"Loop-In-Alarm"	↵	À chaque pression sur une touche, la fonction configurée est exécutée (p.ex. "ActiverVue"). Il est ainsi possible d'appeler une vue contenant les informations sur l'alarme sélectionnée. Acquitte également une alarme à acquittement obligatoire. [1)]

[1)] Uniquement si une fonction est programmée au niveau de l'événement "Loop-In-Alarm" pour l'alarme sélectionnée.

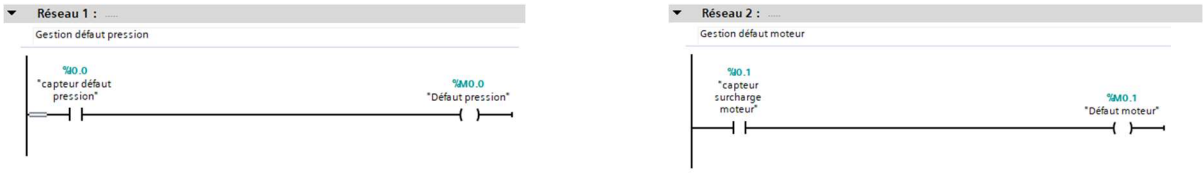
➢ Programmation API (gestion des défauts)

On utilisera un mot de 16 bits (%MW0) qui gèrera les différents défauts. Dans notre exemple, deux défauts (Défaut de pression associé à la variable %M0.0 et Défaut moteur associé à la variables %M0.1)

Déclaration des variables

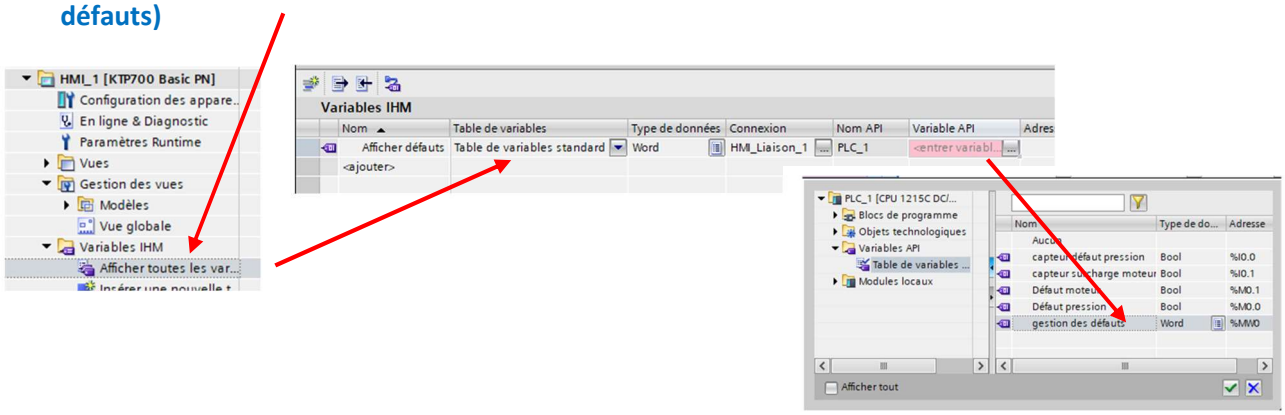


Programme API

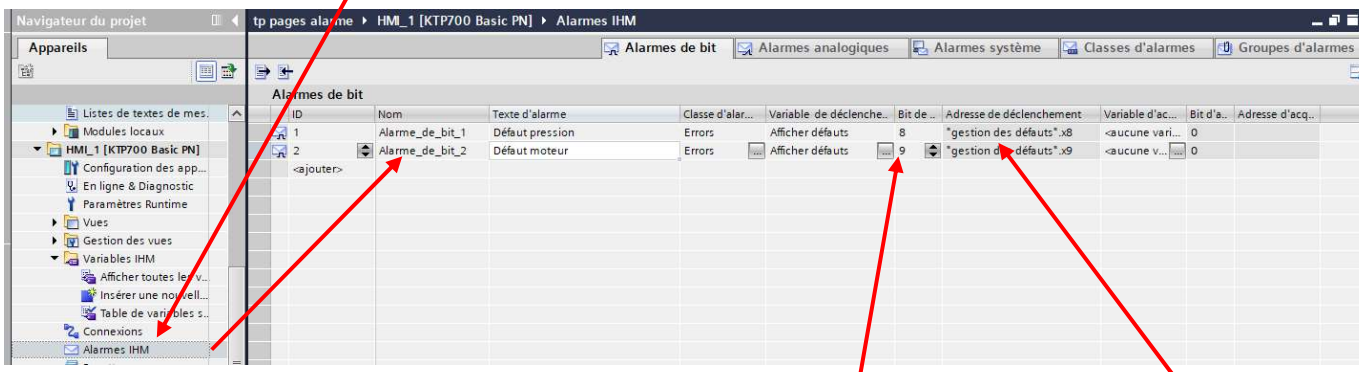


➢ Programmation IHM

Créer une variable IHM (afficher défauts) et faire le lien avec la variable API (gestion des défauts)



- **Créer les différentes alarmes de bit**



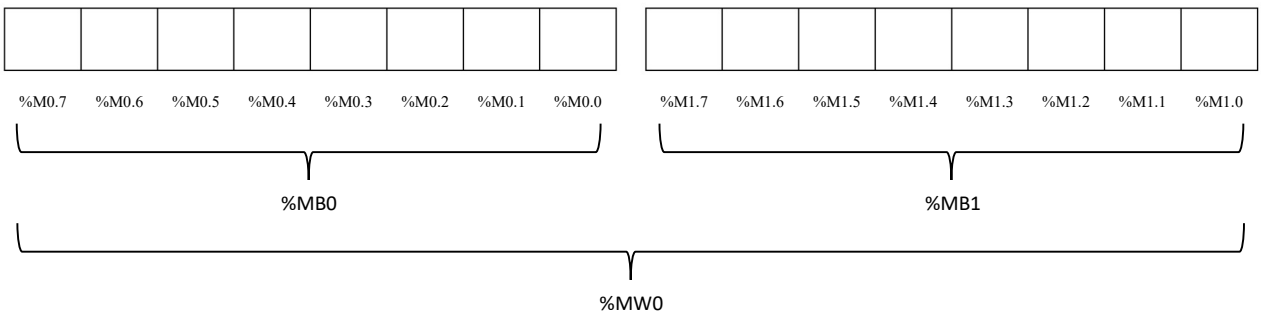
Adresse de déclenchement :

Variable de l'API %MW0

Bit de déclenchement de chaque défaut

Remarques :

Le bit 8 correspond au défaut %M0.0 de la variable API (voir schéma ci-dessous)



Il est possible de mettre du texte d'information sur le défaut détecté.