

# ***Introduction à JDBC***

## ***Manipuler une base de données en Java***

**KABONDO WIYA Jérémie**



**GETSOFT ControlCode**

# Intermédiaire

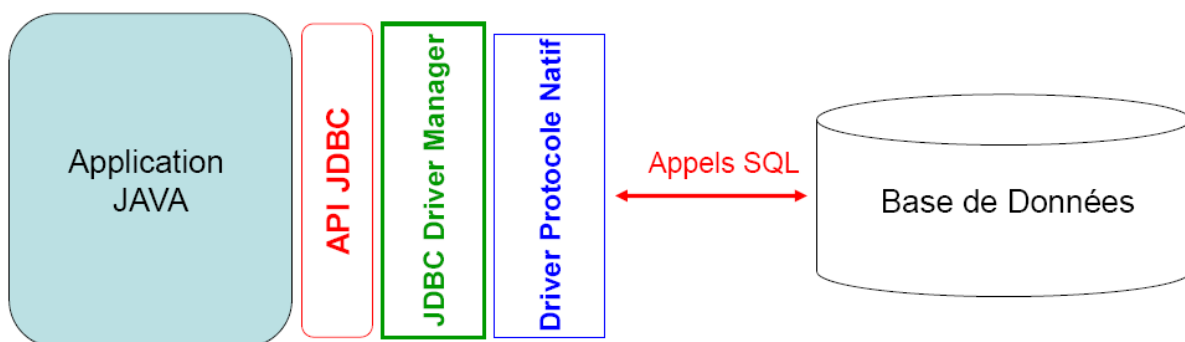
- **JDBC: Java DataBase Connectivity**

- Cette API a été développée par SUN pour permettre à des applications Java d'accéder à des bases de données relationnelles quelconques.

- **Les étapes principales**

- Se connecter à une base de données
- Envoyer une requête SQL
- Manipuler le résultat

- **JDBC: un driver (pilot) fournissant des outils pour ces fonctions**



# Définition

L'API JDBC permet d'exécuter des instructions SQL.

- JDBC fait partie du JDK (Java Development Kit) :
- Paquetage `java.sql`
- `import java.sql.*;`

## Le pilote...

Établit le lien avec la base de données, en sachant "lui parler".

Dans JDBC : des classes chargées de gérer un pilote...

Des pilotes existent pour MySQL, Oracle, PostgreSQL, ACCESS,...

- **JDBC Driver Manager**
  - Driver Protocole Natif
- **La connexion...**
  - Elle peut s'établir si on donne l'adresse de la BD à laquelle se connecter...

# Préparatif

- **Installer un driver JDBC**

- E.g. SQL server 2008 de Microsoft  
<http://msdn2.microsoft.com/en-us/sql/aa336272.aspx>
- Pont ODBC/JDBC (Open DataBaseConnectivity)
- Pour MySQL, recupererle .jar correspondant  
<https://dev.mysql.com/downloads/connector/j/>

## Étape 1: charger le pilote

- **Charger le pilote (driver)**

- Pilote: contient toutes les classes nécessaires pour communiquer avec une base de données
- il faut utiliser la méthode `forName` de la classe `Class`
- E.g.
  - SQL Server 2008:  
`Class.forName("com.microsoft.jdbc.sqlserver.SQLServerDriver");`
  - Pour MySQL 8.0  
`Class.forName("com.mysql.cj.jdbc.Driver");`
- Cette méthode charge en mémoire la classe demandée et exécute son éventuel bloc static.
  - `static{ BaseDriver.registerDriver(new SQLServerDriver()); }`
- Pour que cela fonctionne, il faut définir la variable d'environnement `CLASSPATH` pour inclure le répertoire contenant les classes du driver

## Étape 2: établir une connexion

- Pour établir la connexion avec SQL Server, il faut préciser
  - Le nom de la machine (ou son numéro IP),
  - Le port où le service SQL est démarré (quasiment toujours 1433),
  - Le nom de la base de données,
  - Le login utilisé ainsi que son mot de passe.

```
try { String strClassName = "com.microsoft.jdbc.sqlserver.SQLServerDriver";
    String strUrl = "jdbc:microsoft:sqlserver://hostname:1433;" + "user = sa;
    password = pass; DatabaseName = dbName";
    Class.forName(strClassName);
    Connection conn = DriverManager.getConnection(strUrl);
    // ...
    conn.close();
}
catch(ClassNotFoundException e) {
    System.err.println("Driver non chargé !");
    e.printStackTrace();
} catch(SQLException e) {
    // ...
}
```

## Étape 2: établir une connexion

- Établir la connexion avec MySQL
  - DriverManager: la méthode statique getConnection va créer un objet de connexion
  - La méthode statique getConnections va retourner un objet de connexion (getConnection != getConnections)
  - Paramètre: le protocole et le sous-protocole:
    - jdbc:odbc:DsnName
    - DSN (Data Source Name)

// Declaration des objets

```
private static Connection conn = null;
```

// Methode static pour creer la connection

```
static{
```

```
    try {
```

```
        Class.forName("com.mysql.cj.jdbc.Driver");
```

```
        conn =
```

```
DriverManager.getConnection("jdbc:mysql://localhost:3308/db_biu","root","");
```

```
        // ...
```

```
        conn.close();
```

```
    } catch (ClassNotFoundException | SQLException e) {
```

```
        System.out.println("Database.getConnection() Error --> " +
```

```
e.getMessage());
```

```
        //e.printStackTrace();
```

```
    }
```

```
}
```

// Methode static pour retourner la connection

```
public static Connection getConnections(){
```

```
    return conn;
```

```
}
```

## Étape 3: Requête SQL

- L'exécution d'une requête SQL passe par l'utilisation d'une classe, spécifique au pilote utilisé, implémentant l'interface **Statement**
- Un objet de type **Statement** se doit d'être adapté à la base manipulée. JDBC ne fournit que l'interface Statement, qui est implantée par différentes classes dans un pilote
- Obtenir un objet Statement: avec la méthode **createStatement**.

## Exemple pour s'Authentifier

```
try {
    userName = textUserName.getText();
    passWord = textPassWord.getText();

    if(!userName.equalsIgnoreCase("") && !passWord.equalsIgnoreCase("")) {

        conn = DbConnection.getConnection();
        String query = "SELECT * FROM t_users WHERE utilisateur=? AND
motDePasse=?";
        ptm = conn.prepareStatement(query);
        ptm.setString(1, userName);
        ptm.setString(2, passWord);

        rs = ptm.executeQuery();

        if(rs.next()){
            profil = rs.getString("utilisateur");
            fonction = rs.getString("fonction");

            switch (fonction) {
                case "Administrateur":
                    acl = new TableauDeBord();
                    acl.setVisible(true);
                    acl.LabelUsers.setText(profil);
                    acl.LabelId.setVisible(false);
                    this.dispose();
                    break;
```

```

        case "Assistant":
            acl = new TableauDeBord();
            acl.setVisible(true);
            acl.LabelUsers.setText(profil);
            acl.btnModifier.setVisible(false);
            acl.btnSupprimer.setVisible(false);
            acl.LabelId.setVisible(false);
            this.dispose();
            break;
        default:
            JOptionPane.showConfirmDialog(null, "Votre profil n'est pas
disponible encore !!! ", "Erreur d'authentification !", JOptionPane.ERROR_MESSAGE);
            break;
    }
    }else{
        JOptionPane.showConfirmDialog(null, "Nom utilisateur ou mot de
passe incorrect !!! ", "Erreur d'authentification !", JOptionPane.ERROR_MESSAGE);
    }
    }else{
        JOptionPane.showConfirmDialog(null, "Remplissez les champs svp !!! ",
"Erreur d'authentification !", JOptionPane.ERROR_MESSAGE);
    }

    } catch ( SQLException ex) {
        JOptionPane.showConfirmDialog(null, ex, "Erreur Login !",
JOptionPane.ERROR_MESSAGE);
    }

```



## Exemple pour Ajouter

```
// Methode pour ajouter une personne
// Pour ajouter dans la bd
try {
    if( textNom.getText().equals("") || textPrenom.getText().equals("") ||
textUtilisateur.getText().equals("")
        || textPass.getText().equals("") || cboxFonction.getSelectedIndex()
== -1 || cboxSexe.getSelectedIndex() == -1 ) {

        JOptionPane.showMessageDialog(null, "Entrer les informations
completes SVP ! ");
    }else{

        conn = DbConnection.getConnection();

        pstmt = conn.prepareStatement(" INSERT INTO t_personnel (
nom, prenom, fonction, sexe, utilisateur, motDePasse) VALUES (?, ?, ?, ?, ?, ?); ");
        pstmt.setString(1, textNom.getText().toUpperCase());
        pstmt.setString(2, textPrenom.getText());
        pstmt.setString(3, cboxFonction.getSelectedItem().toString());
        pstmt.setString(4, cboxSexe.getSelectedItem().toString());
        pstmt.setString(5, textUtilisateur.getText());
        pstmt.setString(6, textPass.getText());

        pstmt.executeUpdate();
        pstmt.close();

        ActualiserAffichage();
        JOptionPane.showMessageDialog(null, "Enregistrement effectue
avec succes !!! " );

        Vider();
    }
} catch (SQLException ex) {
    JOptionPane.showMessageDialog(null, "Probleme lors d'insertion du
personnel" + ex.getLocalizedMessage());
}
```

## Exemple pour Afficher

```
// Methode pour afficher toutes les personnes
public void AfficherPersonnelAsc(){
    model.addColumn("matricule");
    model.addColumn("nom");
    model.addColumn("prenom");
    model.addColumn("fonction");
    model.addColumn("sexe");
    model.addColumn("telephone");
    model.addColumn("adresse");

    try {
        conn = DbConnection.getConnection();
        stm = conn.createStatement();
        rs = stm.executeQuery("SELECT * FROM t_personnel
WHERE etat = 0 ORDER BY id DESC");
        while(rs.next()){
            model.addRow(new Object[]{rs.getString("id"),
rs.getString("nom"), rs.getString("prenom"), rs.getString("fonction"),
rs.getString("sexe"), rs.getString("utilisateur"), rs.getString("motDePasse")});
        }
    } catch (SQLException e) {
        JOptionPane.showMessageDialog(null, "Probleme
d'affichage du tableau !" + e.getLocalizedMessage());
    }

    jTable1.setModel(model);
}
```

## Exemple pour Modifier

```
// Methode pour modifier une personne
try{
    // Pour modifier dans la bd
    if( textNom.getText().equals("") || textPrenom.getText().equals("") ||
textUtilisateur.getText().equals("")
    || textPass.getText().equals("") || cboxFonction.getSelectedIndex()
== -1 || cboxSexe.getSelectedIndex() == -1 ) {

        JOptionPane.showMessageDialog(null, "Cliquer sur une ligne
dans le tableau SVP ! ");
    }else{

        if(JOptionPane.showConfirmDialog(null,"Voulez - vous
vraiment modifier","Modification", JOptionPane.YES_NO_OPTION) ==
JOptionPane.OK_OPTION){

            conn = DbConnection.getConnection();
            stm = conn.createStatement();
            String Requete="UPDATE t_personnel SET nom
='"+textNom.getText().toUpperCase()+"', prenom= '"+textPrenom.getText()+"',
fonction = '"+cboxFonction.getSelectedItem().toString()+"', sexe
='"+cboxSexe.getSelectedItem().toString()+"',
utilisateur='"+textUtilisateur.getText()+"', motDePasse='"+textPass.getText()+"'
WHERE id = '"+LabelId.getText()+"' ";
            stm.executeUpdate(Requete);

            ActualiserAffichage();
            JOptionPane.showConfirmDialog(null, "Modification
effectuée avec succes !!! ");

            Vider();
        }
    }
} catch(HeadlessException | SQLException ex){
    JOptionPane.showConfirmDialog(null, "Erreur de modification
!!! " + ex.getLocalizedMessage());
}
```

## Exemple pour Supprimer

```
// Methode pour supprimer definitive une personne dans la BD
public void supprimerPersonne(int idPersonne){
    conn = dbConnexion.getConnection();
    try {

        pstmt = conn.prepareStatement(" DELETE  FROM pers WHERE id =
'" + idPersonne + "' ");

        pstmt.executeUpdate();
        pstmt.close() ;
    } catch (SQLException ex) {
        System.out.println("Erreur sur la suppression definitive de la
personne : " + ex);
    }
}
```

## Exercice Java SWING et MySQL

- Créer une table t\_test (id, fonction, username, password)
- Créer une table t\_personne(matricule, nom, prenom, fonction, genre, telephone, adresse, etat)
- Créer une classe dbConnexion pour la connection
- Créer une classe Java Personnes
- Etablir une connexion MySQL pour :
  - S'authentifier ;
  - Insérer une personne dedans ;
  - Afficher toutes les personnes présentes ;
  - Rechercher une personne ;
  - Modifier une personne ;
  - Supprimer une personne.

- Créer Set up (exécutable) de votre application pour l'installer chez l'utilisateur

## Exemple des formulaires créé avec Swing en Java et une Table en MySQL

***N.B : pendant l'exercice on va créer nos propres formulaires et une table***

Création de la table t\_personne dans la base des données

```
mysql> CREATE TABLE t_personne(  
  -> matricule INT PRIMARY KEY AUTO_INCREMENT,  
  -> nom VARCHAR(50) NOT NULL,  
  -> prenom VARCHAR(50) NOT NULL,  
  -> fonction VARCHAR(50) NOT NULL,  
  -> genre VARCHAR(10) NOT NULL,  
  -> telephone INT(15) NOT NULL,  
  -> adresse VARCHAR(100) NOT NULL,  
  -> etat INT(1) NOT NULL DEFAULT '0');  
Query OK, 0 rows affected, 2 warnings (0.03 sec)  
  
mysql>
```

Formulaire d'authentification



Login

**IDENTIFIEZ - VOUS ICI**

Utilisateur :

Mot de passe :

Connection

## Formulaire d'Insertion et d'affichage des informations

Tableau de bord

Gestion du personnel

Bienvenue admin [Deconnection](#)

Nom

Prenom

Fonction

Sexe

Telephone

Adresse

Rechercher par

[Actualiser](#)

matricule	nom	prenom	fonction	sexe	telephone	adresse
8	IVYARHIMANA	Orly	Personnel	Masculin	76345432	Gihosha
7	AGATTE	AGT	Personnel	Feminin	62345678	Kamenge
6	ASIFIWE	Jolie	Personnel	Feminin	68900922	Bwiza
5	MBONIHAN...	Eriel	Personnel	Masculin	77234343	Zone gihosha
4	MPASA	Odile	Personnel	Feminin	2147483647	Shabunda c...
3	CHEUSI	Elisabeth	Administrate...	Feminin	2147483647	Bukavu
2	NINKURI	Patric	Personnel	Masculin	76223344	Kinama
1	KIKUNI	Jean	Administrate...	Masculin	88324565	Goma

[Ajouter](#) [Modifier](#) [Supprimer](#)

# Création d'un setup et déploiement

Pour notre cas, nous allons utiliser deux applications pour faciliter la création de l'exécutable de notre application développée en Java sous Windows.

## 1. launch4j

Vous pouvez trouver à cette adresse :

<https://sourceforge.net/projects/launch4j/files/launch4j-3/3.11>

Image 1

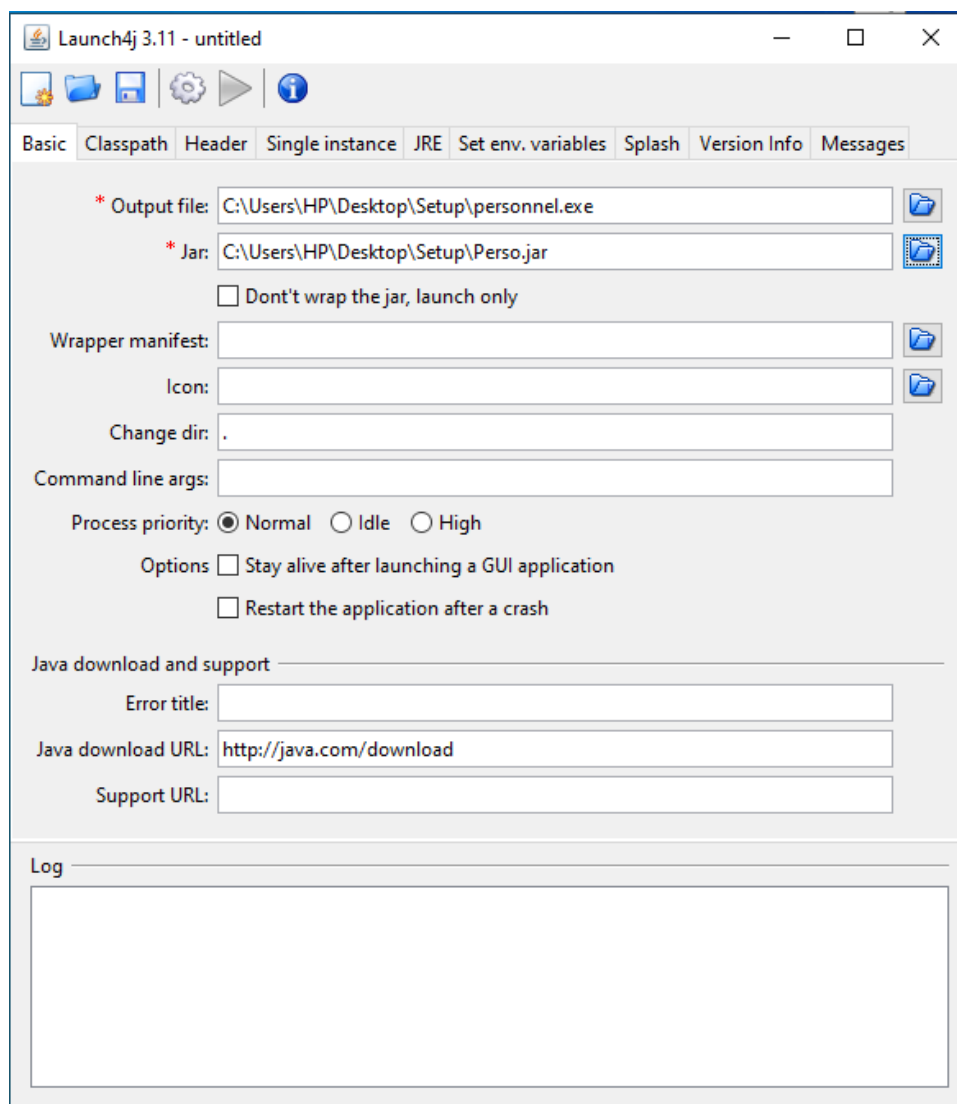


Image 2

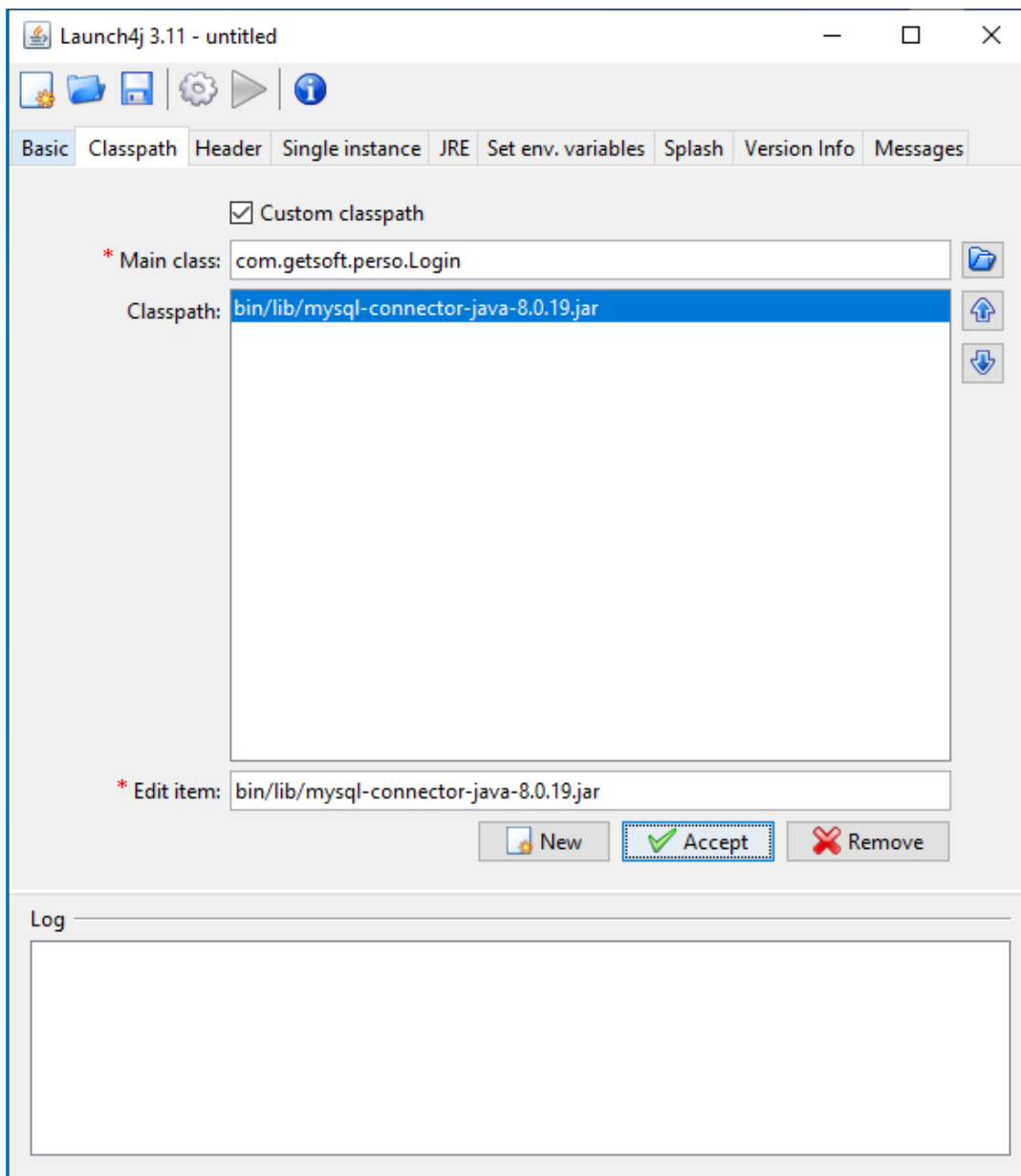




Image 3

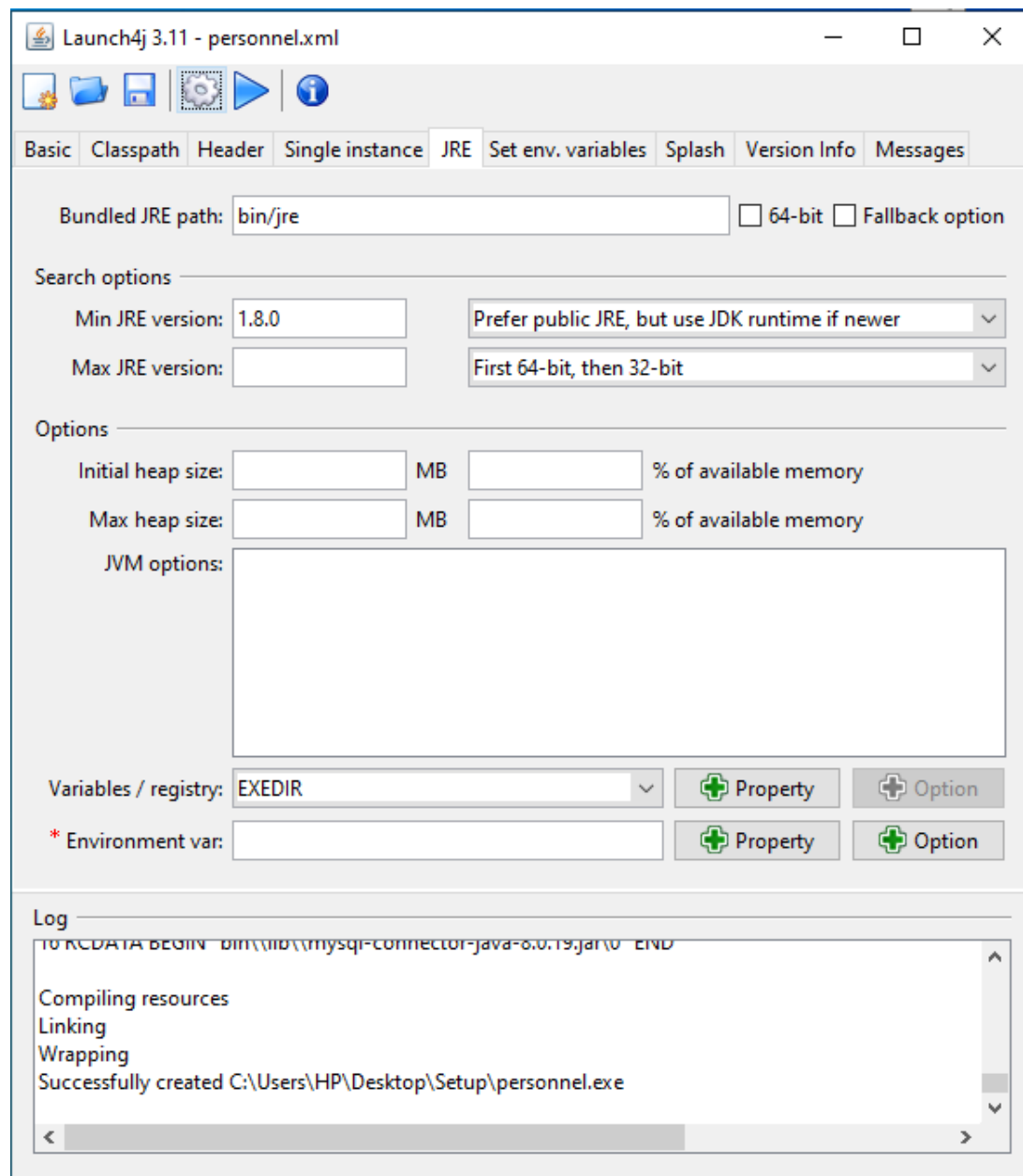


Image 4

The screenshot shows the 'Launch4j 3.11 - personnel.xml' window with the 'Version Info' tab selected. The window has a standard Windows title bar and a toolbar with icons for file operations and help. The 'Version Info' tab contains several sections for configuring application metadata.

**Basic** | **Classpath** | **Header** | **Single instance** | **JRE** | **Set env. variables** | **Splash** | **Version Info** | **Messages**

☒ Add version information

\* File version: 1.0.0.0 \* Free form: nil

\* File description: perso

\* Copyright: getsoft

**Additional information**

\* Product version: 1.0.0.0 \* Free form: nil

\* Product name: appPersonnel

Company name: getsoft

\* Internal name: getsoft

\* Original filename: personnel.exe

Trademarks:

Language: French

**Log**

Compiling resources  
Linking  
Wrapping  
Successfully created C:\Users\HP\Desktop\Setup\personnel.exe

## 2. Inno Setup

Image 1

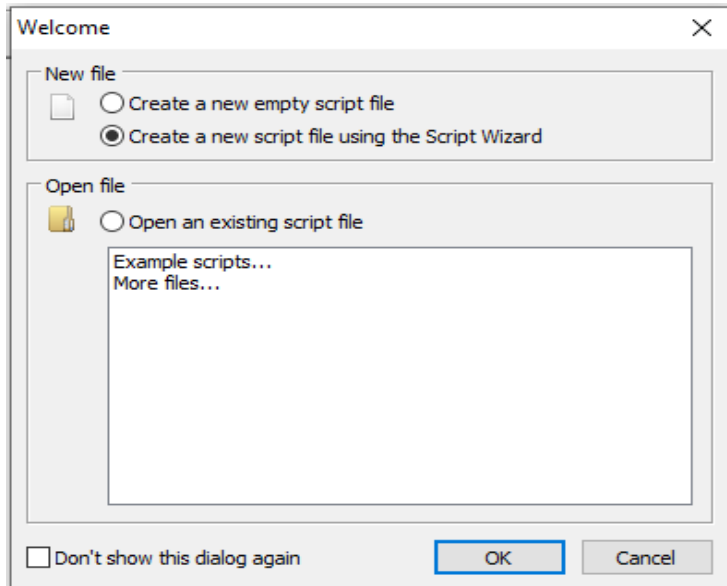


Image 2

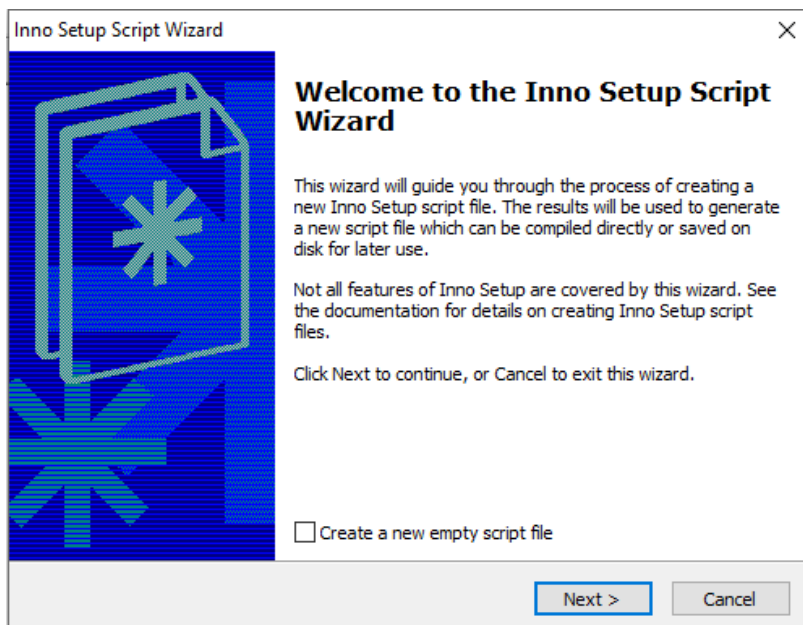
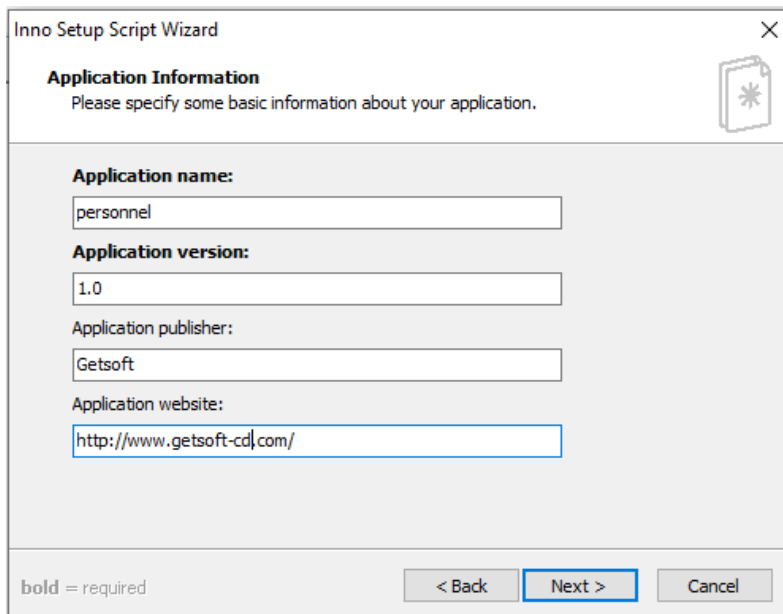


Image 3



The screenshot shows the 'Application Information' screen of the Inno Setup Script Wizard. The window title is 'Inno Setup Script Wizard'. The subtitle is 'Application Information' with a subtext 'Please specify some basic information about your application.' and a document icon with an asterisk. The form contains four text input fields: 'Application name:' with 'personnel', 'Application version:' with '1.0', 'Application publisher:' with 'Getsoft', and 'Application website:' with 'http://www.getsoft-cd.com/'. At the bottom, there is a legend 'bold = required', and three buttons: '< Back', 'Next >' (highlighted with a blue border), and 'Cancel'.

**Application Information**  
Please specify some basic information about your application.

**Application name:**  
personnel

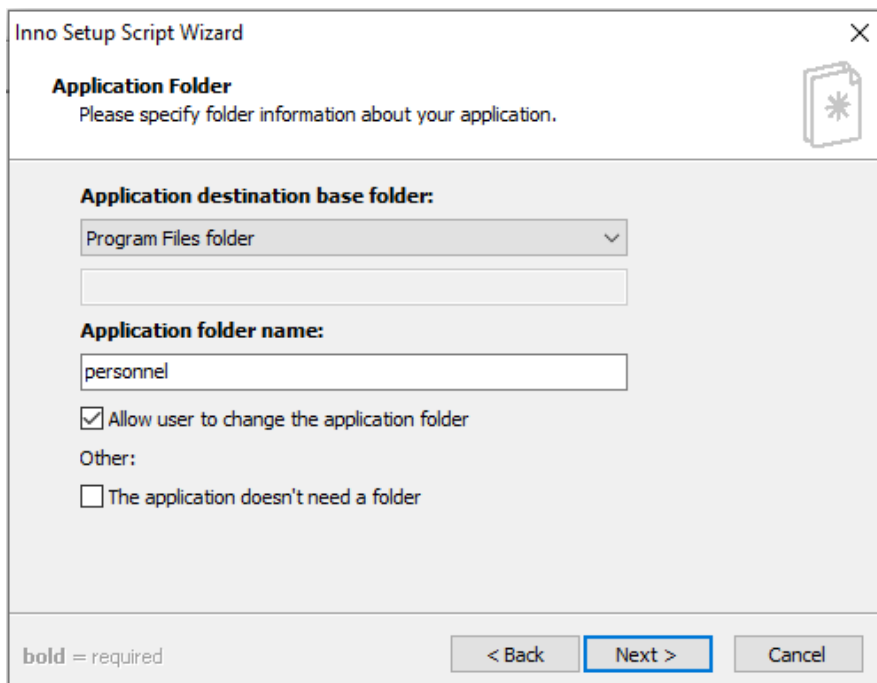
**Application version:**  
1.0

Application publisher:  
Getsoft

Application website:  
http://www.getsoft-cd.com/

bold = required      < Back      Next >      Cancel

Image 4



The screenshot shows the 'Application Folder' screen of the Inno Setup Script Wizard. The window title is 'Inno Setup Script Wizard'. The subtitle is 'Application Folder' with a subtext 'Please specify folder information about your application.' and a document icon with an asterisk. The form contains a dropdown menu for 'Application destination base folder:' set to 'Program Files folder', an empty text field for 'Application folder name:' containing 'personnel', a checked checkbox for 'Allow user to change the application folder', and an 'Other:' section with an unchecked checkbox for 'The application doesn't need a folder'. At the bottom, there is a legend 'bold = required', and three buttons: '< Back', 'Next >' (highlighted with a blue border), and 'Cancel'.

**Application Folder**  
Please specify folder information about your application.

**Application destination base folder:**  
Program Files folder

**Application folder name:**  
personnel

☒ Allow user to change the application folder

Other:  
☐ The application doesn't need a folder

bold = required      < Back      Next >      Cancel

Image 5

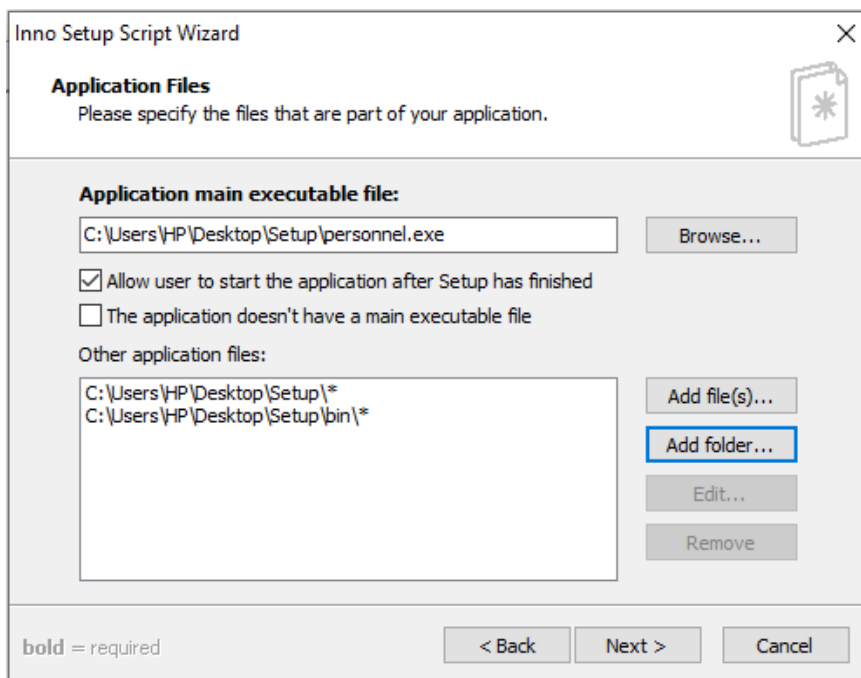


Image 6

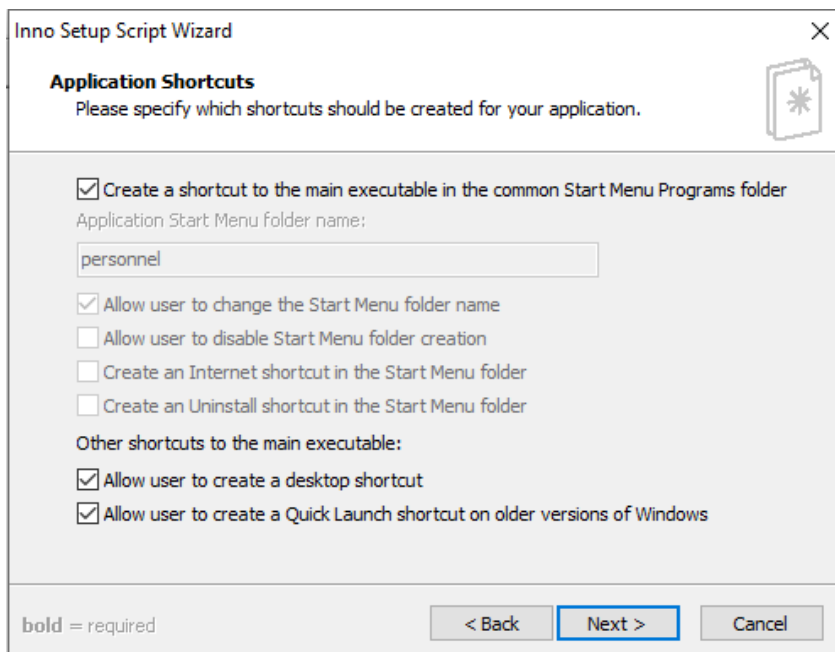


Image 7

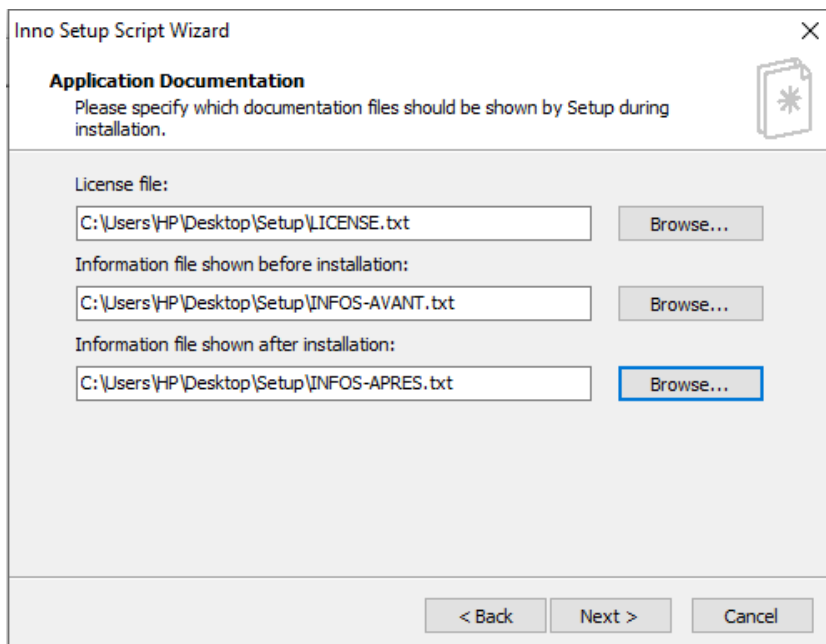


Image 8

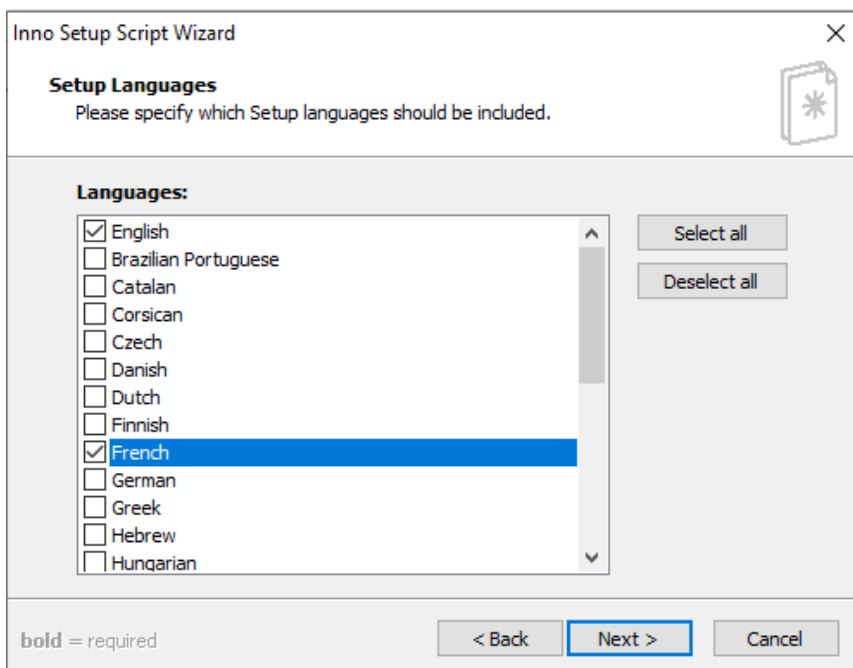
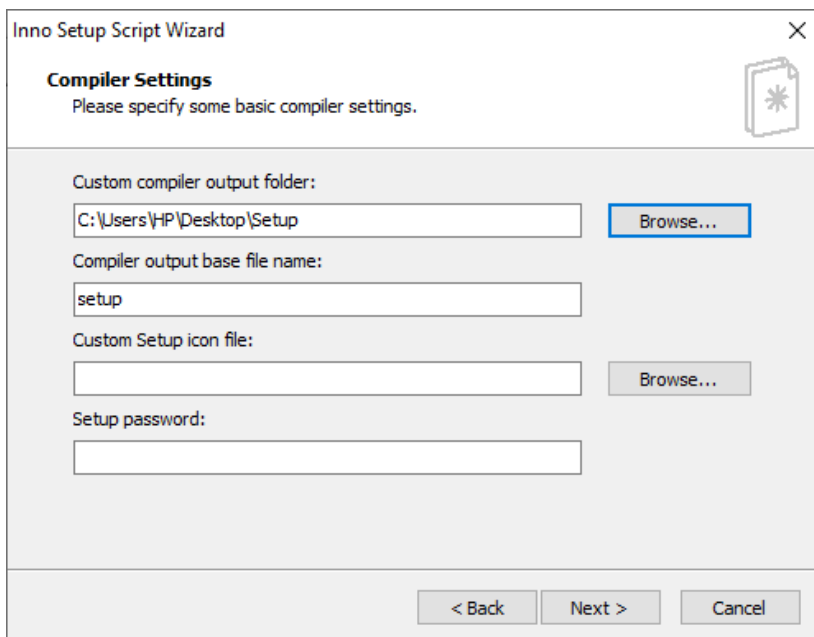


Image 9



The image shows the 'Inno Setup Script Wizard' window, specifically the 'Compiler Settings' step. The window has a title bar with 'Inno Setup Script Wizard' and a close button. Below the title bar is a sub-header 'Compiler Settings' and a message 'Please specify some basic compiler settings.' with a document icon. The main area contains four input fields: 'Custom compiler output folder:' with the text 'C:\Users\HP\Desktop\Setup' and a 'Browse...' button; 'Compiler output base file name:' with the text 'setup'; 'Custom Setup icon file:' with an empty field and a 'Browse...' button; and 'Setup password:' with an empty field. At the bottom are three buttons: '< Back', 'Next >', and 'Cancel'.

Inno Setup Script Wizard

**Compiler Settings**  
Please specify some basic compiler settings.

Custom compiler output folder:  
C:\Users\HP\Desktop\Setup Browse...

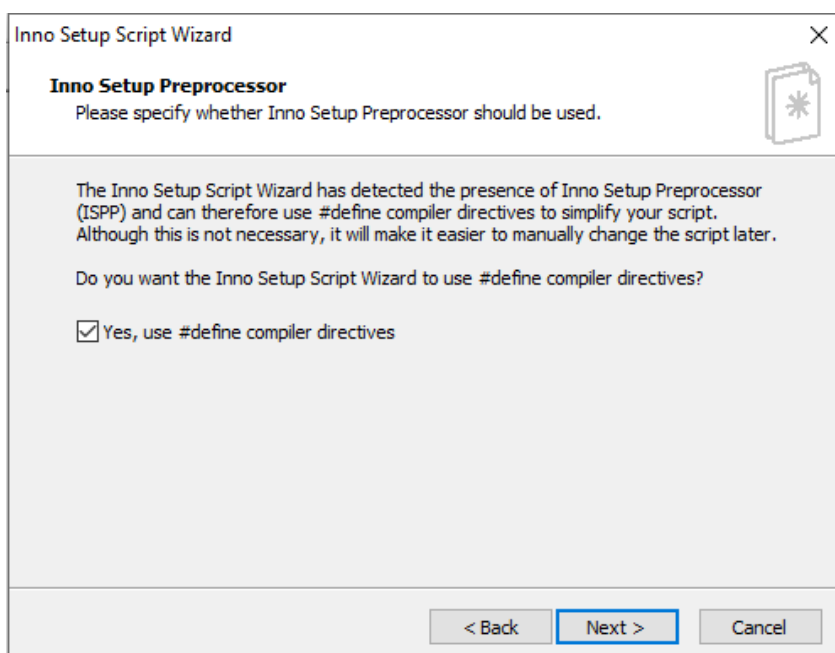
Compiler output base file name:  
setup

Custom Setup icon file:  
Browse...

Setup password:

< Back Next > Cancel

Image 10



The image shows the 'Inno Setup Script Wizard' window, specifically the 'Inno Setup Preprocessor' step. The window has a title bar with 'Inno Setup Script Wizard' and a close button. Below the title bar is a sub-header 'Inno Setup Preprocessor' and a message 'Please specify whether Inno Setup Preprocessor should be used.' with a document icon. The main area contains a paragraph of text explaining the Inno Setup Preprocessor (ISPP) and a question 'Do you want the Inno Setup Script Wizard to use #define compiler directives?'. Below the question is a checked checkbox labeled 'Yes, use #define compiler directives'. At the bottom are three buttons: '< Back', 'Next >', and 'Cancel'.

Inno Setup Script Wizard

**Inno Setup Preprocessor**  
Please specify whether Inno Setup Preprocessor should be used.

The Inno Setup Script Wizard has detected the presence of Inno Setup Preprocessor (ISPP) and can therefore use #define compiler directives to simplify your script. Although this is not necessary, it will make it easier to manually change the script later.

Do you want the Inno Setup Script Wizard to use #define compiler directives?

☒ Yes, use #define compiler directives

< Back Next > Cancel