Les commentaires vont avoir la couleur rouge.

Dans le format BNF, les accolades indiquent que l'expression peut être répétée zéro fois ou plus.  
Par exemple, la règle: <args> :: = <arg> {"," <arg>}   
définit une liste d'arguments conventionnels séparés par des virgules.

Les crochets autour d'une extension, [expansion], indiquent que cette extension est facultative.  
Par exemple, la règle: <terme> :: = ["-"] <facteur>  
permet aux facteurs d'être annulés.

Une expression de la forme <symbole> { <symbole> } est équivalente dans à avoir une expression de la forme (<symbole>)+ qui indique que le symbole entre les parenthèses doit se produire au moins une fois.

J'ai pris comme convention pour la conception de ma grammaire BNF qu'il peut exister deux types de règles de production, ceux qui se développent à des symboles terminaux, et ceux qui se développent à des symboles non terminaux, les premiers vont avoir ses noms en majuscule et les derniers en minuscule (avec quelques exceptions).  
Par exemple, la règle de production <actions> va se développe au symboles non terminaux, alors que <ACTIONS> va se développe au symboles terminaux.

Passons à la grammaire :  
<syntax> ::= <interface> <initializations> <actions>

<interface> ::= <DEBUT> <INTERFACE>

<comp> <prop> <evet>

{ <comp> <prop> <evet> }

<initializations> ::= <INITIALIZATIONS>

<ID > < DOUBLEDOUBLEDOT> <ID > <ASSIGN> (<ID> | <NUMBER > ) <SEMICOLON>

{ <ID > < DOUBLEDOUBLEDOT> <ID > <ASSIGN> (<ID> | <NUMBER > ) <SEMICOLON> }

<actions> ::= <ACTIONS> <expression> { <expression> } <FIN>

<comp> ::= <COMP> <ID> <DOUBLEDOT> <COMPONENT>

[ <OPENPAR> <ID> <CLOSEPAR> ] <SEMICOLON>

<prop> ::= <ID> { <COMMA> <ID> } <DOUBLEDOT>

( <ID> | <OPENTAG> <ID> { <COMMA> <ID> } <CLOSETAG>) <SEMICOLON>

{ <ID> { <COMMA> <ID> } <DOUBLEDOT>

( <ID> | <OPENTAG> <ID> { <COMMA> <ID> } <CLOSETAG>) <SEMICOLON> }

<evet> ::= { <ACTION> [ <OPENPAR> <ID> <CLOSEPAR> ]

<OPENTAG> [ <IF> <ID> ( <EQUAL> | <NOTEQUAL> ) <ID> <THEN> ]

<ID> <ASSIGN> <VALUE> { <OPERATION> <VALUE> } <CLOSETAG> }

<expression> ::= ( <ACTION> <OPENPAR> <ID> <CLOSEPAR> | <OPENTAG> <expression> <CLOSETAG> ) { <NEXTACTION> <expression> }

À partir de là, on passe au règles qui s'étendent aux symboles terminaux dont nous avions besoin pour la grammaire :

<LETTER> ::= "a" | "b" | "c" | "d" | "e" | "f" | "g" | "h" | "i" | "j" | "k" | "l" | "m" | "n" | "o" | "p" | "q" | "r" | "s" | "t" | "u" | "v" | "w" | "x" | "y" | "z" | "A" | "B" | "C" | "D" | "E" | "F" | "G" | "H" | "I" | "J" | "K" | "L" | "M" | "N" | "O" | "P" | "Q" | "R" | "S" | "T" | "U" | "V" | "W" | "X" | "Y" | "Z"

<SEMICOLON> ::= ";"

< DOUBLEDOT> ::= ":"

<DIGIT> ::= "0" | "1" | "2" | "3" | "4" | "5" | "6" | "7" | "8" | "9"

< NUMBER> ::= <DIGIT> {<DIGIT>}

< EQUAL> ::= "="

< NOTEQUAL> ::= "<>"

< DOUBLEDOUBLEDOT> ::= <DOUBLEDOT> <DOUBLEDOT>

< ASSIGN> ::= <DOUBLEDOT> <EQUAL>

< OPENTAG> ::= "{"

< CLOSETAG> ::= "}"

< OPENPAR> ::= "("

< CLOSEPAR> ::= ")"

< COMMA> ::= ","

< IF> ::= "if"

< THEN> ::= "then"

< DEBUT> ::= "DEBUT"

< FIN> ::= "FIN"

< INTERFACE> ::= "INTERFACE"

< INITIALISATIONS> ::= "INITIALISATIONS"

< ACTIONS> ::= "ACTIONS"

< PLUS> ::= "+"

< ACTION> ::= "Click" | "DbClick" | "ChrKeyPress" | "NumKeyPress" | "EntrerKeyPress" | "Drag" | "Drop"

<NEXTACTION> ::= "+" | ";" | "~"

< COMP> ::= "COMP"

< OPERATION> ::= "-" | "/" | "\*"

< EVET> ::= "EVET" | "EVT"

< PROP> ::= "PROP"

<COMPONENT> ::= "Fenêtre" | "boite Dialogue" | "panel" | "Bouton" | "Champs Texte" |

"Combo Box" | "Liste Déroulante" | "case A cochet" | "Bouton Radio" | "Icône"

<ID> ::= <LETTER> {<LETTER>|<NUMBER>}

<VALUE> ::= <ID> | <NUMBER>