

## QCM – Niveau Avancé (PyQt5 & Événements)

### Q1. Que fait le code suivant ?

```
self.button.clicked.connect(lambda: self.label.setText("OK"))
```

- A. Il crée un nouveau signal
  - B. Il appelle la fonction immédiatement
  - C. Il met à jour le QLabel **lors d'un clic**
  - D. Il supprime le texte de la fenêtre
- 

### Q2. Quel signal est émis par un QComboBox lorsqu'on change de sélection ?

- A. valueChanged
  - B. currentChanged
  - C. currentIndexChanged
  - D. selectionChanged
- 

### Q3. Quel est le rôle d'un slot ?

- A. Générer les événements système
  - B. Réagir à un signal
  - C. Créer une fenêtre
  - D. Définir un layout
- 

### Q4. Quelle est la bonne manière de redéfinir un événement clavier ?

- A. def keyEvent(self, event):
  - B. def keyPressEvent(self, event):
  - C. def keyboard(self):
  - D. def pressEvent(self):
- 

### Q5. Quel code ajoute un widget dans une grille (GridLayout) ?

- A. grid.place(widget, row, col)
  - B. grid.addWidget(widget)
  - C. grid.add(row, col, widget)
  - D. grid.addWidget(widget, row, col)
- 

### Q6. Quelle classe permet de dessiner graphiquement dans un widget ?

- A. QPainter
  - B. QDrawable
  - C. QBrush
  - D. QPicture
- 

**Q7. Quel signal permet de détecter qu'une fenêtre est fermée ?**

- A. closed
  - B. destroyed
  - C. Aucun signal : il faut redéfinir closeEvent
  - D. windowClosed
- 

**Q8. Que renvoie self.lineEdit.text() ?**

- A. Un entier
  - B. Un float
  - C. Une chaîne de caractères
  - D. Un tuple
- 

**Q9. Comment arrêter un QTimer ?**

- A. timer.stop()
  - B. timer.kill()
  - C. timer.shutdown()
  - D. timer.exit()
- 

**Q10. Quel est l'ordre logique d'exécution dans PyQt5 ?**

- A. Construction widgets → Boucle événements → Connexions
  - B. Connexions → Construction widgets → Boucle événements
  - C. Construction widgets → Connexions → Boucle événements
  - D. Boucle événements → Connexions → Construction widgets
- 



À toi !  
Envoie tes réponses sous forme :  
1C, 2B, 3A, ...

Je te corrige ensuite.

**Vous avez dit :**

Q1-C; Q2-C; Q3-B; Q4-B; Q5-B; Q6-A; Q7-A; Q8-C; Q9-A; Q10-?

**ChatGPT a dit :**

Voici la **correction détaillée** de ton 2<sup>e</sup> QCM.

Tu progresses très bien, mais ce QCM était plus difficile, donc quelques pièges !

---

 **Correction**

**Q1 — C ✓**

Oui !

Le lambda permet de mettre à jour le texte **lors du clic**, pas avant.

---

**Q2 — C ✓**

Le signal correct d'un QComboBox est :

currentIndexChanged

---

**Q3 — B ✓**

Un **slot** = fonction appelée automatiquement lorsqu'un signal est émis.

---

**Q4 — B ✓**

La bonne méthode pour gérer une touche pressée est :

def keyPressEvent(self, event):

---

**Q5 — B ✗ (bonne réponse : D)**

**Erreur ici :**

En QGridLayout, il faut **spécifier la ligne et la colonne**.

✓ La bonne réponse est :

grid.addWidget(widget, row, col)

Tu as mis **B**, mais addWidget(widget) appartient à QVBoxLayout / QHBoxLayout, pas à QGridLayout.

---

**Q6 — A ✓**

QPainter → dessin 2D dans un widget.

---

**Q7 — A ✗ (bonne réponse : C)**

Attention piège !

Il **n'existe pas** de signal closed dans PyQt.

Pour détecter la fermeture d'une fenêtre, il faut redéfinir :

```
def closeEvent(self, event):
```

Donc la bonne réponse est : **C**

---

**Q8 — C ✓**

lineEdit.text() renvoie une **chaîne de caractères (str)**

---

**Q9 — A ✓**

Arrêter un timer se fait avec :

```
timer.stop()
```

---

**Q10 — ? → bonne réponse : C**

L'ordre logique est :

- 1 Construction des widgets**
- 2 Connexion signaux/slots**
- 3 Boucle d'événements (exec\_())**

Donc la bonne réponse est : **C**

---

**Q1. Quel module contient la classe QWidget ?**

- A. PyQt5.QtCore
- B. PyQt5.QtGui

- C. PyQt5.QtWidgets
- D. PyQt5.CoreWidgets

**Q1 — C ✓**

QWidget se trouve bien dans **PyQt5.QtWidgets**.

---

**Q2. Quelle méthode est appelée lorsqu'un widget est affiché à l'écran ?**

- A. paintEvent
- B. showEvent
- C. displayEvent
- D. visibleEvent

**Q2 — B ✓**

showEvent est appelée quand un widget devient visible

---

**Q3. Quel signal un QSlider émet-il lorsqu'on change sa valeur ?**

- A. valueChanged
- B. sliderChanged
- C. changed
- D. moved

**Q3 — A ✓**

Un slider émet le signal **valueChanged** quand sa valeur change.

---

**Q4. Quelle classe permet de charger une image pour l'afficher dans un QLabel ?**

- A. QPixmap
- B. QPainter
- C. QImageLoader
- D. QIcon

**Q4 — A ✓**

QPixmap sert à charger des images pour un QLabel.

---

#### **Q5. Que fait le code suivant ?**

```
self.timer = QTimer()
```

```
self.timer.start(500)
```

- A. Lance une action toutes les 500 secondes
- B. Lance une action toutes les 500 millisecondes
- C. Lance une action une seule fois après 500 ms
- D. Ne fait rien sans un signal connecté

**Q5 — D ✓**

!! Attention à ce piège :

Un QTimer.start() seul **ne fait rien tant qu'aucun slot n'est connecté au signal timeout**.

Donc D est correcte

---

#### **Q6. Quel layout place automatiquement les éléments de gauche à droite ?**

- A. QVBoxLayout
- B. QHBoxLayout
- C. QFormLayout
- D. QGridLayout

**Q6 — B ✓**

HBoxLayout place les widgets **horizontalement**.

---

#### **Q7. Quel est le rôle de super().\_\_init\_\_() dans une classe dérivée de QWidget ?**

- A. Arrêter la boucle d'événements
- B. Exécuter le constructeur de la classe parente
- C. Créer un signal
- D. Charger les widgets automatiquement

**Q7 — B ✓**

super().\_\_init\_\_() exécute le constructeur de la classe parente (QWidget ici).

---

**Q8. Quel signal est émis par un QLineEdit quand l'utilisateur appuie sur Entrée ?**

- A. returnPressed
- B. enterPressed
- C. pressed
- D. keyPressed

**Q8 — B  (Bonne réponse : A)**

La bonne réponse est :

**✓ A — returnPressed**

Il n'existe pas de signal enterPressed.

---

**Q9. Quelle méthode est utilisée pour fixer la taille d'un widget ?**

- A. setSize()
- B. setDimension()
- C. resize()
- D. setWidthHeight()

**Q9 — C ✓**

resize(width, height) fixe la taille d'un widget.

---

**Q10. Quel widget permet d'afficher une liste simple d'éléments ?**

- A. QTableWidget
- B. QListWidget
- C. QTreeWidget
- D. QLineEdit

**Q10 — B ✓**

QListWidget affiche une **liste simple** d'éléments.



Correction Partie 2 – Code à compléter

---

### Exercice 1

 **Ta réponse :** clicked.connect

 **Correct**

La ligne doit être :

```
self.button.clicked.connect(self.afficher)
```

---

### Exercice 2

 **Ta réponse :** text

 **Correct**, mais il faut bien mettre les parenthèses :

```
texte = self.nomInput.text()
```

---

### Exercice 3 — PRESQUE juste !

 Tu as répondu : timeout 2000

Il manque un morceau dans la première partie.

 **Correction complète :**

```
self.timer.timeout.connect(self.miseAJour)
```

```
self.timer.start(2000)
```

Tu as juste oublié .connect.

---

### Exercice 4

 **Ta réponse :** event

**Correct !**

```
def mousePressEvent(self, event):
```

```
    print("Clic détecté")
```

---

### Exercice 5

👉 Ta réponse : addWidget

Correct !

```
layout.addWidget(self.label)
```