

# Petit guide : la macro \tkzpillbadge

Badges « pilule » (icône ou image + texte), genèse et construction pas à pas

C. Pierquet

Septembre 2026



## Table des matières

|          |                                                                 |           |
|----------|-----------------------------------------------------------------|-----------|
| <b>1</b> | <b>Présentation</b>                                             | <b>2</b>  |
| <b>2</b> | <b>Dépendances</b>                                              | <b>2</b>  |
| <b>3</b> | <b>Genèse : construire la forme</b>                             | <b>2</b>  |
| 3.1      | Le disque et le rectangle à bouts arrondis . . . . .            | 2         |
| 3.2      | Le point de raccord exact (un peu de trigonométrie) . . . . .   | 3         |
| 3.3      | Icône, et le vrai découpage propre . . . . .                    | 4         |
| 3.3.1    | Recadrer le logo, pour éviter les débordements . . . . .        | 4         |
| 3.3.2    | Recadrer le texte, pour éviter les débordements . . . . .       | 4         |
| <b>4</b> | <b>Travail sur les dimensions</b>                               | <b>5</b>  |
| 4.1      | Hauteur : auto / dauto / valeur fixe . . . . .                  | 5         |
| 4.2      | Largeur : auto ou valeur fixe . . . . .                         | 6         |
| 4.2.1    | Largeur fixe : retrouver la longueur du segment droit . . . . . | 6         |
| 4.2.2    | Largeur automatique : mesurer le texte . . . . .                | 6         |
| <b>5</b> | <b>Quelques éléments de personnalisation</b>                    | <b>7</b>  |
| 5.1      | Couleur du contenu : txtcolor . . . . .                         | 7         |
| 5.2      | autoraise : recaler le badge sur la ligne de base . . . . .     | 7         |
| 5.3      | D'autres idées . . . . .                                        | 7         |
| <b>6</b> | <b>La macro finale</b>                                          | <b>8</b>  |
| 6.1      | Le trousseau de clés final . . . . .                            | 8         |
| 6.2      | Les dimensions et styles utilisés . . . . .                     | 8         |
| 6.3      | La macro interne pour le tracé . . . . .                        | 9         |
| 6.4      | La macro principale . . . . .                                   | 10        |
| <b>7</b> | <b>Compléments</b>                                              | <b>11</b> |
| 7.1      | \gentkzpillbadge : fabriquer des badges à texte fixe . . . . .  | 11        |
| 7.2      | Les quatre badges déjà prédéfinis . . . . .                     | 11        |
| 7.3      | Aligner plusieurs badges sur la même largeur . . . . .          | 12        |
| <b>8</b> | <b>Récapitulatif</b>                                            | <b>13</b> |

# 1 Présentation

`\tkzpillbadge` est une macro  $\text{\LaTeX}$ /TikZ, intégrée dans `customenvs-tikzbadge`, qui affiche un **badge en forme de pilule** : un disque (icône `tablericons`, ou image personnalisée) accolé à une capsule arrondie contenant un texte – le style classique des étiquettes `TRUE/FALSE`.

- hauteur **automatique** (calée sur la police courante) ou **fixe**;
- largeur **automatique** (calculée à partir du texte) ou **fixe** (largeur totale imposée);
- couleur du disque et de la capsule (`color`), et couleur du contenu (`txtcolor`) totalement indépendantes;
- icône `tablericons`, ou **image personnalisée** en lieu et place, avec découpe circulaire optionnelle (`imageclip`);
- une macro `\gentkzpillbadge` pour fabriquer des badges à texte figé;
- quatre badges **prédéfinis** prêts à l'emploi :
  - `\tkzpillbadgevrai`,
  - `\tkzpillbadgefaut`,
  - `\tkzpillbadgetrue`,
  - `\tkzpillbadgefals`.

## Remarque

Ce guide ne redécrit pas la macro finale d'un bloc : il retrace les **étapes de construction**, avec les hésitations, essais et corrections qui ont mené à la version publiée dans `customenvs-tikzbadge.sty`. Chaque section correspond à une difficulté concrète rencontrée en chemin.

# 2 Dépendances

## Code

```
\RequirePackage{customenvs-macros}%simplekv, graphics, tikz, halignmakebox...
\RequirePackage{tablericons}%ifthen, calc, graphicx, xstring, simplekv
```

## Remarque

Une première version ne chargeait que `tikz` et `tablericons`, en misant sur le fait que `tablericons` embarque déjà `simplekv/ifthen/graphicx` en interne. Mais les badges **prédéfinis** (§7.2) utilisent en propre `\halignmakebox`, qui elle vient bien de `customenvs-macros` – autant assumer cette dépendance explicitement plutôt que de dupliquer localement le code de `\halignmakebox` pour l'éviter (voir la discussion en §7.3).

# 3 Genèse : construire la forme

## 3.1 Le disque et le rectangle à bouts arrondis

Premier réflexe : `rounded corners` suffit pour un rectangle arrondi simple, mais on a vite besoin de contrôler **indépendamment** le rayon du demi-cercle gauche (le futur disque) et celui de droite. On part donc directement de `arc`, pour maîtriser chaque demi-cercle séparément :

## Code

```
\draw[fill=green!55!black,draw=none]
  (-0.1,-0.5r) circle[radius=0.5] ; % disque
\draw[fill=green!55!black,draw=none]
  (0,0) -- ++(1,0)
  arc (90:-90:r) % demi-cercle droit (passe par l'est, 0°)
  -- ++(-6,0)
  arc (-90:90:r) ; % demi-cercle gauche (passe par l'est, 0°)
```

## Sortie



## Piège rencontré

Visuellement, l'arrondi gauche du rectangle ne *doit* pas avoir le même rayon que l'arrondi droite : l'idée est de considérer que le bord gauche du rectangle est *bloqué* par un disque de rayon légèrement plus grand que le disque rempli !

Le rendu visuel doit faire penser à un *anneau* !

## 3.2 Le point de raccord exact (un peu de trigonométrie)

Le vrai disque de l'icône a un rayon 0.5, mais l'ensemble disque+gap (offset) définit un cercle **de référence** de rayon  $R = 0.5 + \text{offset}$ , sur lequel démarre l'arc gauche de la capsule. Le point de départ de cet arc est l'intersection entre ce cercle et la droite  $y = 0.5$  (le haut du badge) :

$$R \sin \theta = 0.5 \implies \theta = \arcsin\left(\frac{0.5}{R}\right), \quad x_{\text{dep}} = \sqrt{R^2 - 0.25}$$

## Code

```
\pgfmathsetmacro{\R}{0.5+\tkzbadgehoffset}  
\pgfmathsetmacro{\thetaDeg}{asin(0.5/\R)}  
\pgfmathsetmacro{\xdep}{sqrt(\R*\R - 0.25)}
```

L'arc gauche de la capsule se referme alors entre  $-\theta$  et  $+\theta$ , sur le cercle de rayon  $\R$  :

## Code

```
\fill[couleur] (0,0) circle[radius=0.5] ;  
\fill[couleur] ({\xdep},0.5) --++ ({\tkzbadgelen},0) arc(90:-90:0.5)  
--++ (-\tkzbadgelen,0) arc(-\thetaDeg : \thetaDeg : \R) ;
```

## Sortie



## Remarque

`\tkzbadgelen` est la longueur du **segment droit** de la capsule (pas la largeur totale) – toute la section 4.2 lui est consacrée.

### 3.3 Icône, et le vrai découpage propre

#### 3.3.1 Recadrer le logo, pour éviter les débordements

L'empilement (§ 3.2) suffit pour un disque **plus grand**, mais dès que le contenu du disque (icône, image) doit rester strictement à l'intérieur du cercle – sans jamais déborder sur la capsule –, il faut un vrai clip :

##### Code

```
\fill[couleur] (0,0) circle[radius=0.5] ;
\begin{scope}
  \clip (0,0) circle[radius=0.5] ;
  \node[txtcolor,inner sep=0pt] at (0,0)
    {\tablericon[height=0.85cm]{\tkzbadge\logo}} ;%
\end{scope}
```

##### Sortie



##### À retenir

Par défaut, l'icône du disque est dépendante de `tablericons`, mais la bascule icône/image se fait sur **une seule clé**, `image` : vide par défaut (donc `\tablericon` est utilisé), sinon le contenu de la clé est affiché tel quel – un simple `\includegraphics[...]{...}` suffit, aucune macro dédiée n'est nécessaire côté utilisateur.

#### 3.3.2 Recadrer le texte, pour éviter les débordements

Une image (souvent carrée) ne remplit pas naturellement un disque – ses coins peuvent déborder du cercle du badge. La clé booléenne `cliplogo` ajoute un second clip, cette fois circulaire, uniquement pour la branche « image » :

##### Code

```
\begin{scope}
  \ifboolKV[t kz badge]{cliplogo}%
    {\clip (0,0) circle[radius=0.45] ;}{}%
  \node[{\tkzbadge\textcolor},inner sep=0pt] at (0,0) {\tkzbadgeimage} ;
\end{scope}
```

##### Remarque

Désactivée par défaut (`cliplogo=false`) : une image déjà ronde, ou un logo avec de la marge, n'a pas besoin d'être recadrée – la découpe supplémentaire n'est activée que sur demande explicite.

Même logique de clip pour le texte, contenu **dans** la capsule uniquement :

##### Code

```
\fill[couleur] ({\xdep},0.5) --++({\tkzbadgelen},0) arc(90:-90:0.5)
  --++(-\tkzbadgelen,0) arc(-\thetaDeg:\thetaDeg:\R) ;
\begin{scope}
  \clip ({\xdep},0.5) --++({\tkzbadgelen},0) arc(90:-90:0.5)
    --++(-\tkzbadgelen,0) arc(-\thetaDeg:\thetaDeg:\R) ;
  \node[txtcolor,t kz badge\text] at ({\xdep+\tkzbadgelen/2+0.25},0) {<label>} ;
\end{scope}
```

## 4 Travail sur les dimensions

### 4.1 Hauteur : auto / dauto / valeur fixe

Trois façons de fixer la hauteur totale du badge, pour qu'il s'insère proprement dans du texte courant quelle que soit sa taille de police :

#### Code

```
\ifthenelse{\equal{\tkzbadgeheight}{dauto}}%
  {%hauteur ET profondeur d'un texte de référence (ex. "qM")
   \setttotalheight\tkzbadgeautoheight{\hbox{\tkzbadgesdtrut}}%
   \settodepth\tkzbadgeautodepth{\hbox{\tkzbadgesdtrut}}%
   \pgfmithsetmacro{\tkzbadgescale}{\tkzbadgeautoheight/1cm}%
  }%
{%
  \setlength\tkzbadgeautodepth{0pt}%
  \ifthenelse{\equal{\tkzbadgeheight}{auto}}%
    {%uniquement la hauteur, sans jambage (ex. "M")
     \setttotalheight\tkzbadgeautoheight{\hbox{\tkzbadgestrut}}%
     \pgfmithsetmacro{\tkzbadgescale}{\tkzbadgeautoheight/1cm}%
    }%
    {%dimension fixe donnée par l'utilisateur
     \pgfmithsetmacro{\tkzbadgescale}{\tkzbadgeheight/1cm}%
    }%
  }%
```

Le rayon du disque valant 0.5 en coordonnées locales (diamètre 1, soit 1cm à scale=1), le facteur d'échelle du tikzpicture se déduit directement du rapport entre la hauteur voulue et cette référence de 1cm :

#### Code

```
\begin{tikzpicture}[scale=\tkzbadgescale,transform shape,...]
\end{tikzpicture}
```

#### Piège rencontré

L'alignement vertical du badge sera gérée par une éventuelle surcouche dépendante de \raisebox.

## 4.2 Largeur : auto ou valeur fixe

### 4.2.1 Largeur fixe : retrouver la longueur du segment droit

Si l'utilisateur donne une `width` (largeur **totale** du badge, bord gauche du disque au bord droit de la capsule), il faut en retrancher les parties géométriques fixes (`xdep`, et les deux rayons 0.5) pour isoler `\tkzbadgelen` :

$$\text{tkzbadgelen} = \frac{\text{width}}{\text{scale}} - x_{\text{dep}} - 1$$

#### Code

```
\setlength\tkzbadgewidthcalc{\tkzbadgewidth}%  
\pgfmathsetmacro{\tkzbadgelen}{\tkzbadgewidthcalc/1cm/\tkzbadgescale - \xdep - 1}%
```

#### Piège rencontré

La division par `\tkzbadgescale` est **indispensable** : `width` est une dimension physique absolue (5cm), qui ne doit pas être affectée par le `scale` du dessin – il faut donc « annuler » par avance la mise à l'échelle à venir. Un premier essai sans cette division donnait une largeur physique fausse dès que `height=auto/dauto` produisait un `scale` différent de 1.

### 4.2.2 Largeur automatique : mesurer le texte

En `width=auto`, on mesure la largeur du texte **déjà redimensionné** à la hauteur cible, qui est gérée par le nombre décimal (`\tkzbadgeratioheight`), en dehors du `tikzpicture` :

#### Code

```
\settowidth\tkzbadgewidthcalc{%  
  \hbox{\tikz\node[tikzbadgetext]  
    {\resizebox*{!}{\tkzbadgeratioheight}{<texte>}};  
}%  
\pgfmathsetmacro{\tkzbadgelen}{\tkzbadgewidthcalc/1cm + \R/2 - \xdep/2}%
```

#### Remarque

Ici, **pas** de division par `\tkzbadgescale` : la mesure et `\tkzbadgelen` sont toutes deux en unités locales, et seront multipliées par le même facteur au moment du tracé – la division se simplifie donc d'elle-même. Le terme `+\R/2 - \xdep/2` est une marge empirique, ajustée pour que le texte ne touche jamais l'arc droit.

#### À retenir

L'utilisation d'un `\vphantom` (référence `qM` par défaut) garantit que **tous** les textes sont redimensionnés à la **même** hauteur de référence, quels que soient leurs jambages/accents – sans lui, deux badges `VRAI` et `Qui ?` n'auraient pas exactement la même hauteur visuelle de texte.

## 5 Quelques éléments de personnalisation

### 5.1 Couleur du contenu : `txtcolor`

Icône et texte étaient d’abord câblés en blanc, ce qui ne fonctionne que sur fond de page blanc. Une clé dédiée, appliquée aux deux node, suffit à rendre le badge lisible sur n’importe quel fond :

#### Code

```
\node[\tkzbadge\textcolor,inner sep=0pt] at (0,0) {...} ; % icône/image
\node[\tkzbadge\textcolor,\tkzbadge\text] at (... ,0) {...} ;% texte
```

### 5.2 `autoraize` : recaler le badge sur la ligne de base

Par défaut, le badge doit se comporter comme un caractère inline, centré sur la ligne de base du texte environnant – pas posé par son bas. La clé booléenne `autoraize` pilote un `\raisebox` conditionnel :

#### Code

```
\ifboolKV[\tkzbadge]{autoraize}{%
  {\raisebox{-\tkzbadgeautodepth}{\inttkzbadgedraw{#2}{#3}}}%
  {\inttkzbadgedraw{#2}{#3}}}%
```

#### Remarque

`autoraize=false` reste utile pour composer plusieurs badges **à la main** dans un `tikzpicture` englobant (par exemple pour les faire pivoter individuellement), où c’est le bas du badge, pas son centre, qu’on veut aligner sur une référence commune.

### 5.3 D’autres idées

D’autres éléments de personnalisations *fin*es ont été envisagées et implémentées, comme le fait de pouvoir spécifier :

- la hauteur relative du texte dans le cartouche ;
- le décalage horizontal entre le disque et le cartouche ;
- les caractères de base pour l’*alignement* du badge ;
- les caractères de base pour le positionnement du texte du badge.

## 6 La macro finale

### 6.1 Le trousseau de clés final

#### Code

```
\defKV[tkzbadge]{%
  color      = \def\tkzbadgecolor{#1},%
  txtcolor   = \def\tkzbadgetxtcolor{#1},%
  txtratio   = \def\tkzbadgetxtratio{#1},%
  height     = \def\tkzbadgeheight{#1},%
  icon       = \def\tkzbadgelogo{#1},%
  width      = \def\tkzbadgewidth{#1},%
  hoffset    = \def\tkzbadgehoffset{#1},%
  strut      = \def\tkzbadgestrut{#1},%
  dstrut     = \def\tkzbadgesdtrut{#1},%
  image      = \def\tkzbadgeimage{#1},%
  txtphantom = \def\tkzbadgevphant{#1}%
}
\setKVdefault[tkzbadge]{%
  color      = gray,
  txtcolor   = white,
  txtratio   = 0.8,
  height     = dauto,
  icon       = check,
  width      = auto,
  hoffset    = 0.075,
  strut      = {M},
  dstrut     = {qM},
  autoraise  = true,
  image      = {},
  txtphantom = {qM},
  cliplogo   = false
}
```

| Clé            | Rôle                                    | Défaut |
|----------------|-----------------------------------------|--------|
| color          | couleur du disque et de la capsule      | gray   |
| txtcolor       | couleur de l'icône/image et du texte    | white  |
| txtratio       | hauteur locale du texte (référence 1cm) | 0.8    |
| icon           | nom d'icône tablericons                 | check  |
| image          | remplace icon si non vide               | {}     |
| height         | auto / dauto / dimension fixe           | dauto  |
| width          | auto / dimension fixe (largeur totale)  | auto   |
| hoffset        | écart disque/capsule                    | 0.075  |
| strut / dstrut | référence hauteur (auto/dauto)          | M / qM |
| txtphantom     | référence hauteur du texte affiché      | qM     |
| autoraise      | booléen, recentrage vertical            | true   |
| cliplogo       | booléen, découpe circulaire de l'image  | false  |

### 6.2 Les dimensions et styles utilisés

#### Code

```
\newlength\tkzbadgeautoheight
\newlength\tkzbadgeautodepth
\newlength\tkzbadgewidthcalc
\newlength\tkzbadgeratioheight
\newlength\tkzbadgeimageheight
\setlength\tkzbadgeimageheight{0.85cm}

\tikzset{tkzbadgetext/.style={font=\sffamily\bfseries,text depth=0pt,inner sep=0pt}}
```

### 6.3 La macro interne pour le tracé

#### Code

```
\newcommand\inttkzbadgedraw[3][]{%
% #1 = inutilisé // #2 = options tikzpicture // #3 = label
\begin{tikzpicture}[scale=\tkzbadgescale,{\tkzbadgecolor},#2,transform shape]
\fill (0,0) circle[radius=0.5] ;
\begin{scope}%scope left circle
\clip (0,0) circle[radius=0.5] ;
\begin{scope}%scope small left circle if cliplogo=true
\ifboolKV[tikzbadge]{cliplogo}%
{\clip (0,0) circle[radius=0.5\tkzbadgeratioheight] ;}{}%
\ifthenelse{\equal{\tkzbadgeimage}{}}{%
{%
\node[{\tkzbadgetxtcolor},inner sep=0pt] at (0,0)
{\tablericon[height=1.075\tkzbadgeratioheight]{\tkzbadgeicon}} ;
}%
}%
\node[{\tkzbadgetxtcolor},inner sep=0pt] at (0,0) {\tkzbadgeimage} ;
}%
\end{scope}
\end{scope}
\fill ({\xdep},0.5) --++ ({\tkzbadgelen},0) arc(90:-90:0.5)
--++ (-\tkzbadgelen,0) arc(-\thetaDeg : \thetaDeg : \R) ;
\begin{scope}%scope right rectangle circle
\clip ({\xdep},0.5) --++ ({\tkzbadgelen},0) arc(90:-90:0.5)
--++ (-\tkzbadgelen,0) arc(-\thetaDeg : \thetaDeg : \R) ;
\node[{\tkzbadgetxtcolor},tkzbadgetext] at ({\xdep+\tkzbadgelen/2+0.25},0)
{\resizebox*{!}{\tkzbadgeratioheight}{\vphantom{\tkzbadgevphant}#3}};
\end{scope}
\end{tikzpicture}%
}
```

## 6.4 La macro principale

### Code

```
\NewDocumentCommand\tkzpillbadge{ O{} D<>{} m }{%
  \restoreKV[tkzbadge]%
  \setKV[tkzbadge]{#1}%
  \setlength\tkzbadgeratioheight{\tkzbadgetxratio cm}%
  \ifthenelse{\equal{\tkzbadgeheight}{dauto}}{%
    {%
      \setttotalheight\tkzbadgeautoheight{\hbox{\tkzbadgesdtrut}}%
      \settodepth\tkzbadgeautodepth{\hbox{\tkzbadgesdtrut}}%
      \pgfmithsetmacro{\tkzbadgescale}{\tkzbadgeautoheight/1cm}%
    }%
  }%
  {%
    \setlength\tkzbadgeautodepth{0pt}%
    \ifthenelse{\equal{\tkzbadgeheight}{auto}}{%
      {%
        \setttotalheight\tkzbadgeautoheight{\hbox{\tkzbadgestrut}}%
        \pgfmithsetmacro{\tkzbadgescale}{\tkzbadgeautoheight/1cm}%
      }%
    }%
    \pgfmithsetmacro{\tkzbadgescale}{\tkzbadgeheight/1cm}%
  }%
}%
\pgfmithsetmacro{\R}{0.5+\tkzbadgehoffset}%
\pgfmithsetmacro{\thetaDeg}{asin(0.5/\R)}%
\pgfmithsetmacro{\xdep}{sqrt(\R*\R - 0.25)}%
\ifthenelse{\equal{\tkzbadgewidth}{auto}}{%
  {%
    \settowidth\tkzbadgewidthcalc{%
      \hbox{%
        \tikz\node[tkzbadgetext]
          {\resizebox*{!}{\tkzbadgeratioheight}{\vphantom{\tkzbadgevphant}{#3}}};
      }%
    }%
    \pgfmithsetmacro{\tkzbadgelen}{\tkzbadgewidthcalc/1cm + \R/2 - \xdep/2}%
  }%
  {%
    \setlength\tkzbadgewidthcalc{\tkzbadgewidth}%
    \pgfmithsetmacro{\tkzbadgelen}%
      {\tkzbadgewidthcalc/1cm/\tkzbadgescale - \xdep - 1}%
  }%
}%
\ifboolKV[tkzbadge]{autoraise}%
  {\raisebox{-\tkzbadgeautodepth}{\inttkzbadgedraw{#2}{#3}}}%
  {\inttkzbadgedraw{#2}{#3}}%
}
```

## 7 Compléments

### 7.1 \gentkzpillbadge : fabriquer des badges à texte fixe

Une fois un badge réglé, \gentkzpillbadge génère une nouvelle macro autonome, texte figé, mais toujours personnalisable ponctuellement :

#### Code

```
\NewDocumentCommand\gentkzpillbadge{ m m m }{%  
  % #1 = macro -- #2 = options -- #3 = texte  
  \NewDocumentCommand{#1}{ 0{} D<>{} }{%  
    \tkzpillbadge[#2,##1]<##2>{#3}%  
  }%  
}
```

#### Code

```
\gentkzpillbadge{\monbadgevrai}{color=green!50!black,icon=check}{VRAI}  
\gentkzpillbadge{\monbadgefaux}{color=red!50!black,icon=x}{FAUX}
```

#### Sortie



#### À retenir

Le double \NewDocumentCommand imbriqué impose de **doubler** les # de la commande interne (##1, ##2) : #1–#3 appartiennent à \gentkzpillbadge et sont déjà substitués au moment où le corps de la nouvelle macro est écrit ; ##1–##2 appartiennent à la macro **générée**, lue seulement à son propre appel, plus tard.

### 7.2 Les quatre badges déjà prédéfinis

customenvs-tikzbadge fournit d'office quatre badges construits sur ce principe – c'est justement leur code qui a motivé le retour à la dépendance customenvs-macros (voir la remarque de la section 2) :

#### Code

```
\NewDocumentCommand\tkzpillbadgevrai{}{%  
  \tkzpillbadge[color=green!50!black]{\halignmakebox{VRAI}{VRAI,FAUX}}%  
}  
\NewDocumentCommand\tkzpillbadgefaux{}{%  
  \tkzpillbadge[color=red!50!black,icon=x]{\halignmakebox{FAUX}{VRAI,FAUX}}%  
}  
\NewDocumentCommand\tkzpillbadgetrue{}{%  
  \tkzpillbadge[color=green!50!black]{\halignmakebox{TRUE}{TRUE,FALSE}}%  
}  
\NewDocumentCommand\tkzpillbadgefalse{}{%  
  \tkzpillbadge[color=red!50!black,icon=x]{\halignmakebox{FALSE}{TRUE,FALSE}}%  
}
```

#### Sortie



#### Remarque

VRAI/FAUX ont la même longueur (4 lettres) : `width=auto` seul les aurait déjà alignés naturellement. Le couple `TRUE(4)/FALSE(5)`, en revanche, a vraiment besoin de `\halignmakebox` pour partager la même largeur.

En plaçant les options prédéfinies (#2) **avant** celles de l'appel (##1), l'utilisateur final peut surcharger ponctuellement une clé du préreglage (dernière valeur d'une clé qui l'emporte).

### 7.3 Aligner plusieurs badges sur la même largeur

`width=auto` calcule la largeur de **chaque** badge indépendamment de son propre texte – des textes de longueurs différentes donnent donc des largeurs différentes. Pour que plusieurs badges partagent la largeur du plus long du groupe, on combine `width=auto` avec `\halignmakebox` de `customenvs-macros` (chargé en interne) :

#### Code

```
\tkzpillbadge[color=pink!50!black] {\halignmakebox{True} {True,False,Really,Cool}}
\tkzpillbadge[color=teal!50!black] {\halignmakebox{False} {True,False,Really,Cool}}
\tkzpillbadge[color=orange!50!black]{\halignmakebox{Really}{True,False,Really,Cool}}
\tkzpillbadge[color=lime!50!black] {\halignmakebox{Cool} {True,False,Really,Cool}}
```

#### Sortie

✓ True  
✓ False  
✓ Really  
✓ Cool

#### Remarque

`\halignmakebox{texte}{liste}` place `texte` dans une boîte dont la largeur est celle du **plus long élément** de `liste` – ici `Really`. Les quatre badges se retrouvent donc alignés sur cette même largeur, au lieu de se contracter chacun sur son propre texte.

## 8 Récapitulatif

| Commande / concept                                  | Rôle                                       |
|-----------------------------------------------------|--------------------------------------------|
| <code>\tkzpillbadge[clés]&lt;tikz&gt;{texte}</code> | le badge lui-même                          |
| <code>\gentkzpillbadge{macro}{clés}{texte}</code>   | fabrique un badge à texte figé             |
| <code>\tkzpillbadgevrai/faux/true/false</code>      | 4 badges prêts à l'emploi                  |
| <code>imageclip</code>                              | découpe circulaire optionnelle d'une image |
| <code>autoraïse</code>                              | gérer l'alignement vertical                |

- `simplekv` : <https://ctan.org/pkg/simplekv>
- `tablericons` : <https://ctan.org/pkg/tablericons>
- `tikz/pgf` : <https://ctan.org/pkg/pgf>
- `démo/code` : <https://w.blablatex.com/cIJ20nc>