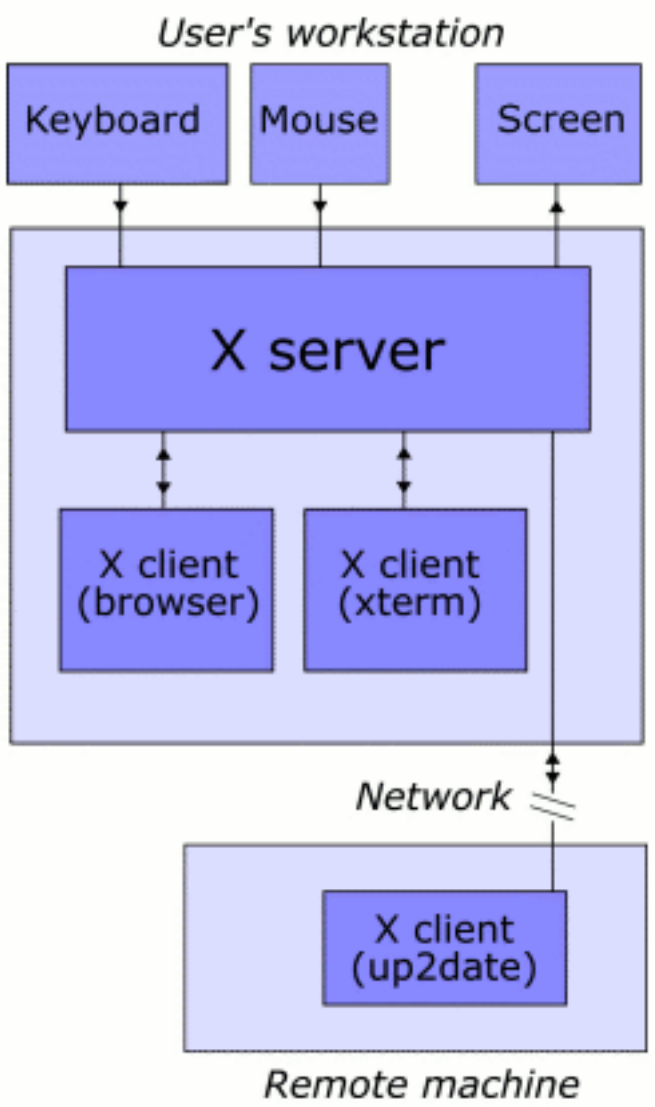
XWindow : l'Environnement Graphique d'UNIX

X Window résumé

- Système de fenêtrage (*windowing*)
  - Pour périphérique d'affichage bitmap
  - Utilisé sur Unix, Unix-like et OpenVMS
  - Disponible sur la majorité des OS
- Fournit les **bases** (minimales) d'un environnement graphique
  - Dessin et déplacement de fenêtres
  - Gestion des interactions (souris, clavier)
  - **Ne** s'occupe **pas** de l'interface utilisateur
    - A la charge de programmes utilisateurs complémentaires
- Offre la transparence réseau (modèle client/serveur)
  - programmes (clients) et affichage (serveur) sur machines

Le modèle client/serveur de X11

- Le serveur
  - Programme qui contrôle le matériel
  - Reçoit des requêtes d'affichage (venant des clients)
  - Envoie des évènements aux clients
- Le client
  - Le code de l'application graphique
  - Peut s'exécuter
    - sur la même machine que le serveur
    - sur une autre machine
- Le protocole (X version 11 : X11)
  - format des données entre client et serveur



Comment le client trouve-t-il le serveur ?

- Un client qui démarre doit trouver un serveur pour ses affichages !
  - Affichage local ?
  - Affichage sur une autre machine ? Laquelle ?
- Un client = un processus = **variables d'environnements**
  - variable réservée pour XWindow : DISPLAY
  - format :
    - DISPLAY=[machine.domaine]:num_serv[.écran]
    - Ex:
      - DISPLAY=toto.titi.org:0.0 : ecran 0, serveur 0 sur toto.titi.org
      - **DISPLAY=:0.0 : ecran 0, serveur 0 sur localhost**
      - DISPLAY=:0 : serveur 0 (écran 0) sur localhost
      - DISPLAY=:0.1 : écran 1 sur serveur 0 de localhost
      - DISPLAY=:1.0 : écran 0 sur serveur 1 de localhost

Sécurité de la connexion entre client et serveur ?

- Mécanismes d'authentification
  - X11-MIT-MAGIC-COOKIE (actif par défaut)
  - Principe
    - le serveur est initialisé avec un clef secrète (COOKIE)
      - au moment du lancement de la session
    - la clef est stockée dans un fichier : $HOME/.Xauthority
      - commande xauth permet de décrypter le fichier
    - le fichier n'est lisible que par l'utilisateur propriétaire du serveur
      - propriétaire = celui qui a démarré la **session** X11
      - tous les programmes (et onc les clients) de cet utilisateur ont le droit de lire le fichier
    - Pour prouver sa bonne foi, un client doit présenter la clef au serveur
      - Totalement transparent si client et serveur s'exécutent sur la même machine

Sécurité de communication entre client et serveur ?

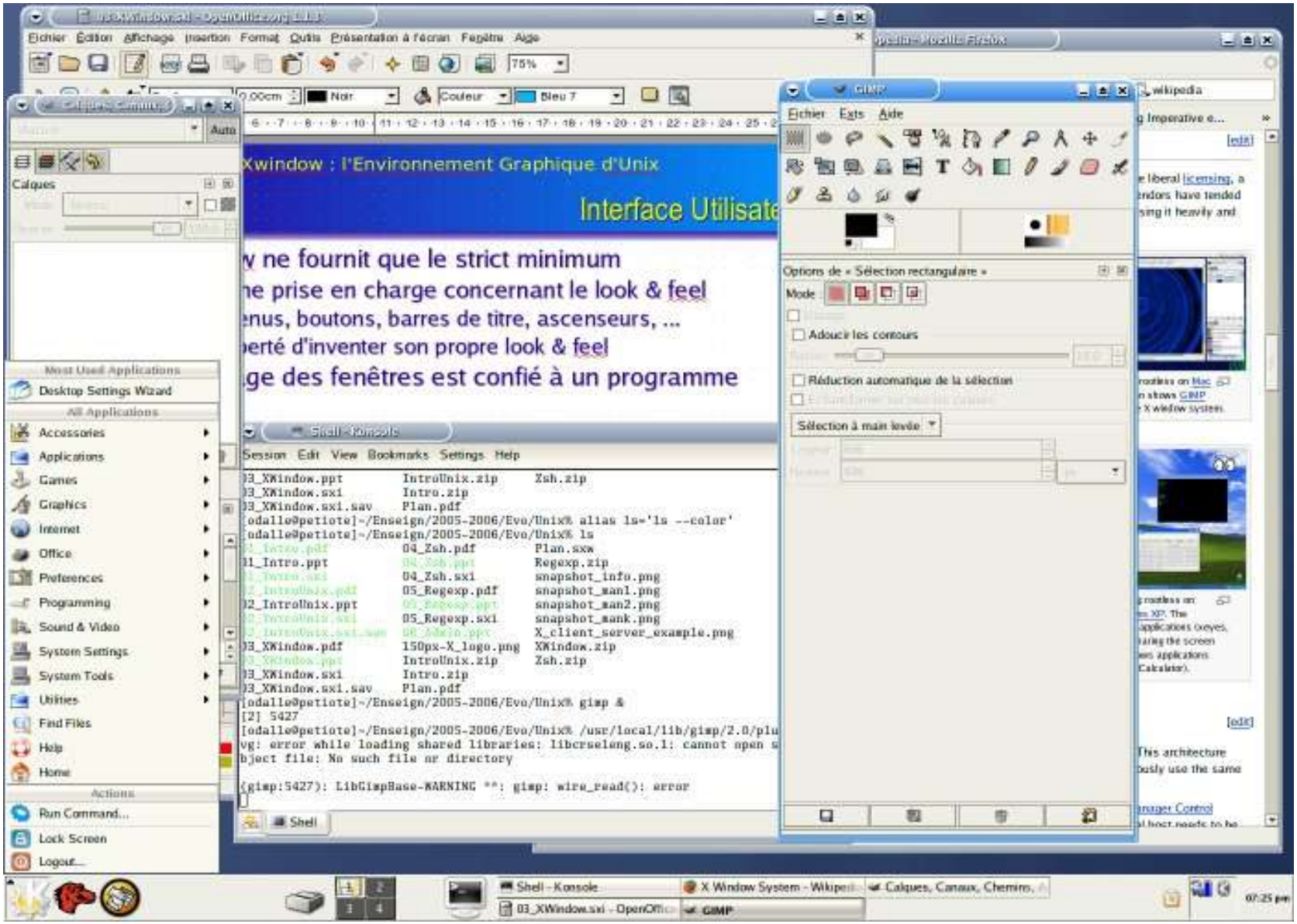
- Communication **non** cryptée
  - Aucun risque si client et serveur sur la même machine
  - Risque : client et serveur sur machine différente
    - Les messages du protocole X11 circulent en clair ...
      - Donc les mots de passe tapés au clavier...
- Solution pour éliminer le risque : ssh
  - ssh permet la création de « tunnel » crypté entre 2 machine
    - Support simplifié pour X11
    - Exemple typique :
      - Client sur machine distante + Serveur local

1. Configurer DISPLAY : `export DISPLAY=:0.0`
2. Connexion ssh : `ssh –X odalle@mascotte.unice.fr`
3. Lancement du client (via ssh, sur la machine distante)

Interface Utilisateur

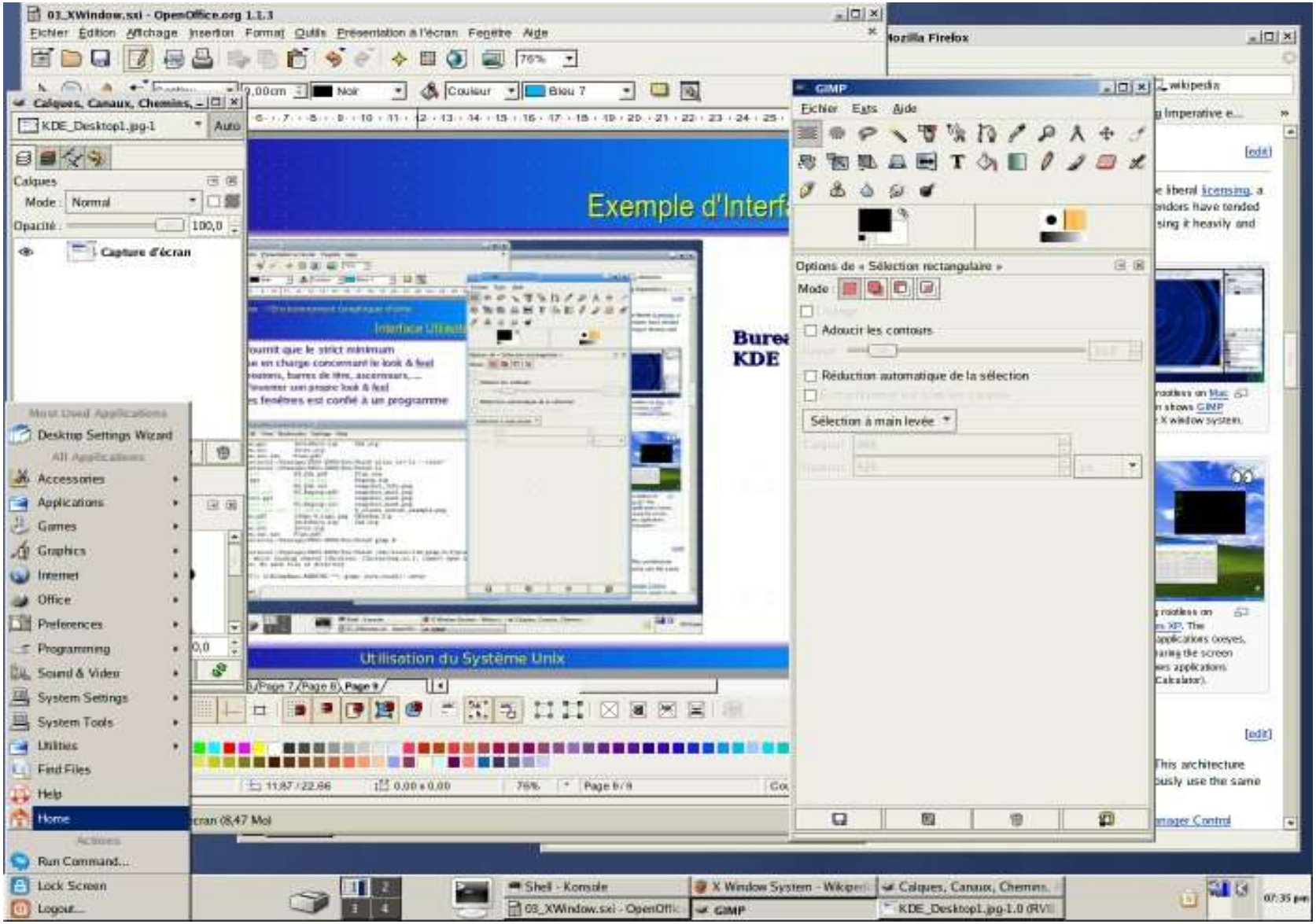
- Xwindow ne fournit que le strict minimum
  - Aucune prise en charge concernant le look & feel
    - menus, boutons, barres de titre, ascenseurs, ...
    - Liberté d'inventer son propre look & feel
- L'habillage des fenêtres est confié à un programme externe
  - Le Window Manager
    - Nombreuse implémentations (= alternatives !)
- On trouve aussi des environnement de bureau
  - plus complets
    - CDE : l'environnement Unix standard
    - Gnome, KDE : les environnements usuels de linux
      - Compatibles avec plusieurs window managers ...

Exemple d'Interface



Bureau KDE

Autre Exemple



Bureau KDE (style windows)

Implémentations

- Une longue histoire ...
  - Projet démarré dans les années 80
  - Pilotage par le « X Consortium »
- Implémentation de référence : X.org
  - Suite de X11R6.6
  - Licence **très** libre
    - Code gratuit, sans obligation de publier les modifications
    - Nombreuses versions dérivées
      - Certaines libres (Xfree, la plus utilisée aujourd'hui)
      - D'autres propriétaires (Sun, DEC/HP, ...)
- Et PC/Windows ? Mac ?
  - Nombreuses version sur windows (Cygwin, eXceed, ...)
  - Mac OS X inclut Apple X11 (dérivé de XFree)

Qu'est-ce qu'un « Terminal X » ?

- Un ordinateur minimal (thin client)
  - Clavier + souris + écran + réseau
  - Exécute uniquement le programme du Serveur X
  - Offre une interface de connexion vers un serveur
    - Propose une liste de serveur disponible sur le réseau
    - Attention à la confusion ...
      - Le « Serveur X » du terminal offre une liste de « Serveurs »
      - Les « Serveurs » (d'application) sont de **vraies** machines
  - Une fois que le serveur est sélectionné ...
    - Session normale (Login/password)
    - Tous les clients tournent sur le « Serveur »
    - L'affichage est dirigé vers le Terminal X
      - cad son Serveur X
- Terminal X en voie de disparition ...

Configuration des clients

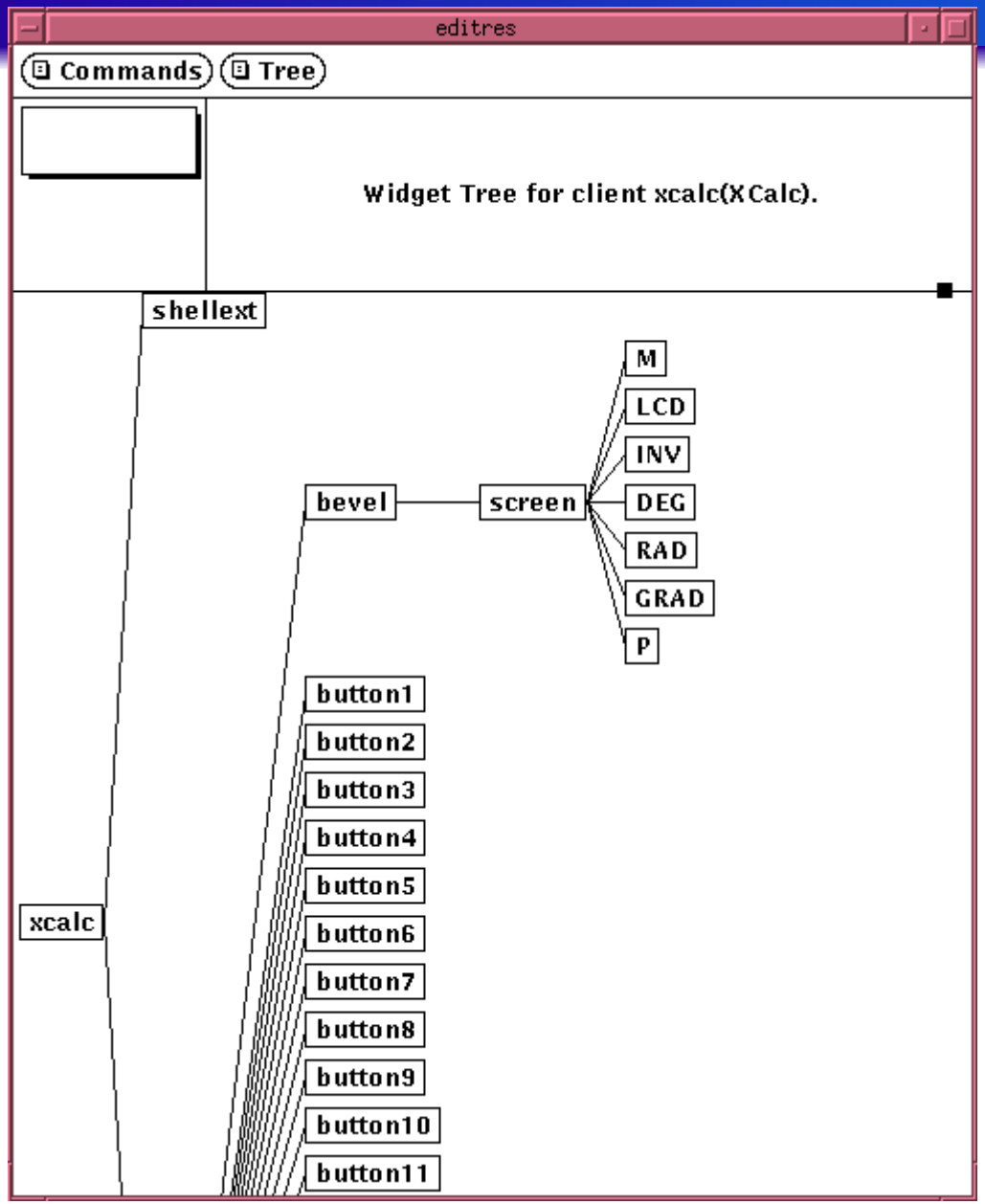
- Ressource : attribut de configuration des objets X11
  - Tous les objets (widget) en ont **beaucoup** !
    - Couleur
    - Géométrie
    - Police
    - Contenu des boutons
    - Taille des éléments emboîtés
    - ...
  - Les objets d'un client sont hiérarchisés
    - Désignation non ambigüe des ressources
      - Par le nom d'instance des objets
      - Par leur classe
    - Ex : la couleur de caractère du bouton sqrt de la calculette...

Exemples de désignation de ressources

- Chemins d'instance
  - `xc.calculator.pave_numérique.zero.labelString : 0`
  - `xc.calculator.pave_numerique.un.labelString : 1`
- Chemins de classe
  - `XC.XmForm.XmForm.XmPushButton.XmString : 0`
  - `XC.XmForm.XmForm.XmPushButton.XmString : 1`

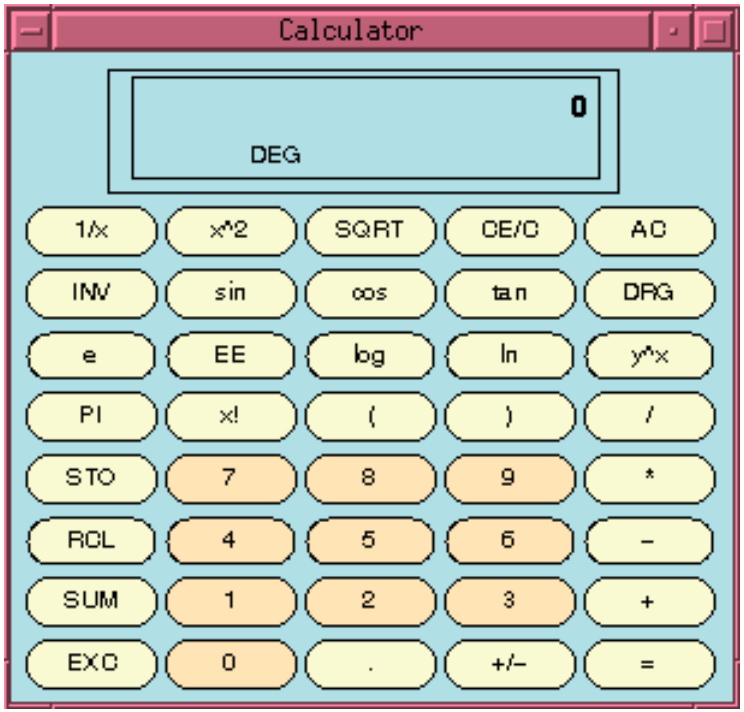
  => Bouton « 0 » étiqueté « 1 » !!
- Chemin mixte
  - `XC.XmForm.XmForm.zero.labelString : 0`
  - `XC.XmForm.XmForm.un.XmString : 1`

Outils pour manipuler les ressources

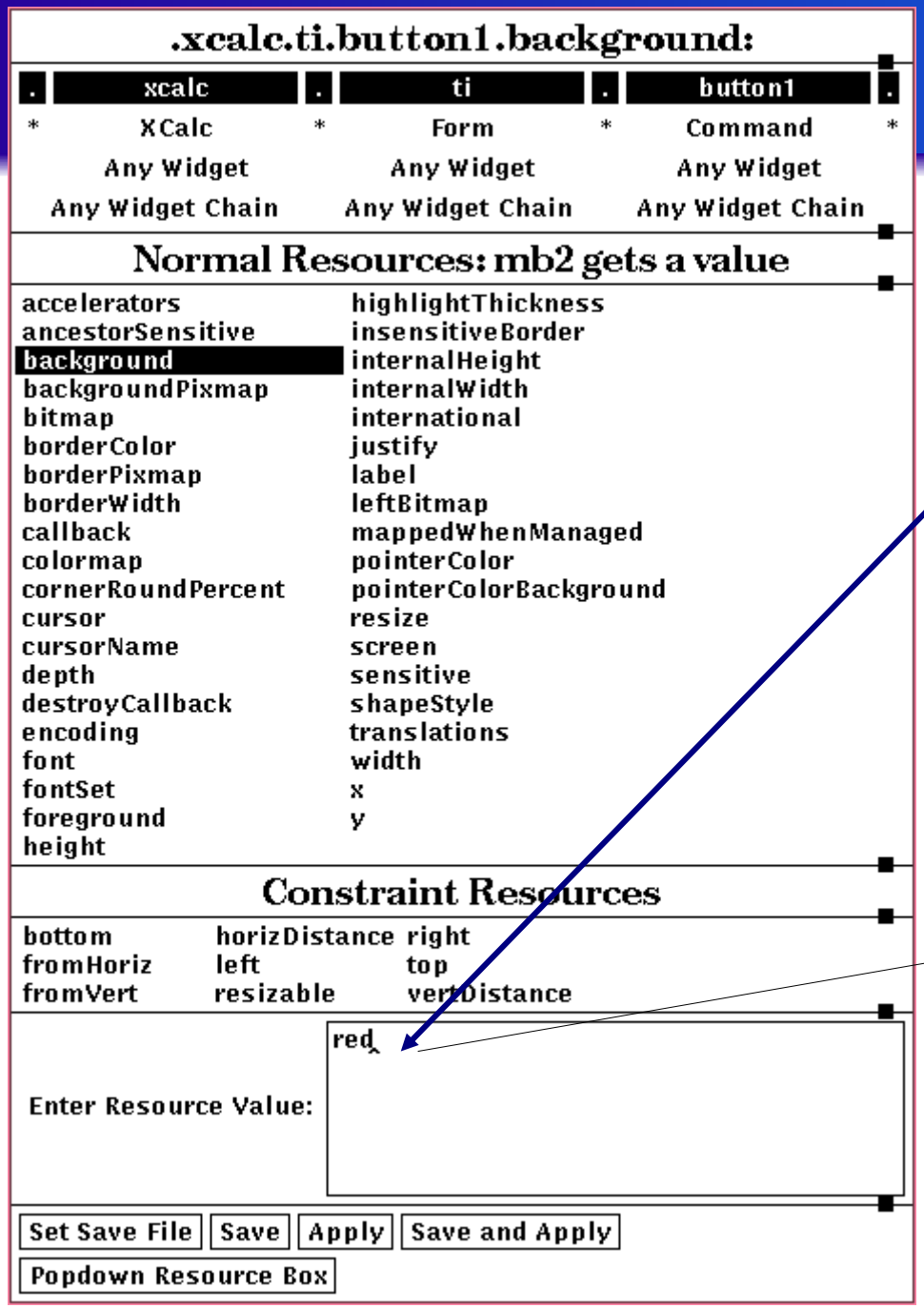


- viewres
- editres

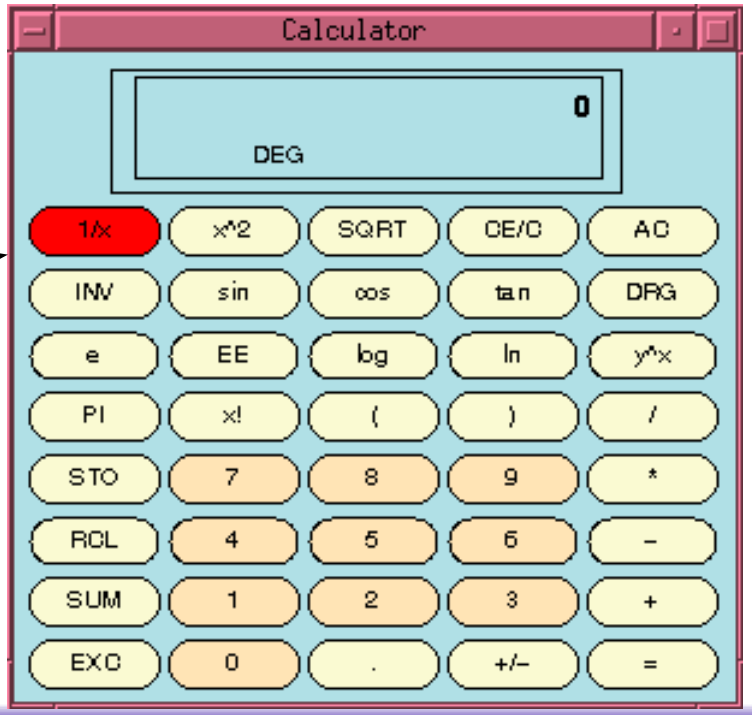
- Ex: calculette



Exemple de modification (editres)



Possibilité de modifier dynamiquement une ressource avec editres



Où sont définies les ressources ?

- Gestionnaire de ressources X (XRM)
  - Décide au moment du lancement du client
  - Décide quelle valeur affecter à chaque ressource
    - Plusieurs sources existent
    - Consultation dans un ordre précis, avec écrasement :
      en cas d'égalité, c'est la dernière spécification qui est gardée
- Sources de définition
  1. Valeur par défaut pour le site
  2. Valeur par défaut pour l'application de l'utilisateur
  3. Valeurs par défaut de l'utilisateur
  4. Valeurs par défaut de l'utilisateur sur une machine
  5. Paramètres ou options -xrm de la ligne de commande

Paramètres usuels en ligne de commande

- -display *display*
- -geometry *geomtry*
  - `xterm -geometry 80x25+10-10`
- -font *font-spec*
- -fg *color*
- -bg *color*
- -iconic
- -xrm *resource-spec*
  - *`xterm -xrm "xterm*background: blue"`*

La ressource « geometry »

- Définit la taille et la position d'une fenêtre
- [**L** x **H**] [{**+**/**-**} **X** {**+**/**-**} **Y**]
  - L,H : en pixels ou en caractères, selon client
  - +/- X : gauche/droite
  - +/- Y : haut/bas
- Exemples :
  - xterm -geometry 80x25
  - xclock -geometry 80x25+10-20
  - xeyes -geometry +500+1
  - xfontsel -geometry -1-1