Konzept KeyStrokes

## CRM Anforderungen

1. Sichten Wechsel mit den Funktionstasten -> diese Keys müssen von überall her getriggert werden können, auch wenn sich der Fokus in einem Feld auf einem Form befindet. Die Keys müssen deshalb direkt auf dem Desktop registriert sein.
   * F3 Arbeitssicht
   * F4 Reportsicht
   * F6 Adminsicht
2. Die Suche muss direkt über die Funktionstaste F2 erreicht werden können.
3. Toolbuttons müssen über die Funktionstasten erreichbar aus der ganzen Applikation erreichbar sein, auch wenn der Fokus zur Zeit in einem Feld liegt.
4. Steuerung des OutlineTrees muss von überall her möglich sein. Dazu wird die Default Tastenbelegung des Tree widgets mit einer Modifier-Taste verwendet z.B. ctrl+ArrowUp
5. Steuerung von OutlineTables auch ohne das sich der Fokus explizit in der Tabelle befindet.
6. Tabwechsel mit automatisch generierten Nummer-KeyStrokes
7. (Processbuttons in Forms mit automatisch generierten KeyStrokes die ebenfalls global gültig sind)

## Keystrokes in CRM gelöst

Die Punkte 1. , 2., und 3. sollen direkt als KeyStrokes auf dem CRM-Desktop beim Erstellen der Sichten, Suche und der Toolbuttons erstellt werden. Dazu wird keine Änderung am Scout vorgenommen.

## Resultierende Anforderungen an Scout

1. Um die Punkte 4., 5. und 6. zu gewährleisten ist einen Anpassung an Scout nötig und zwar sollen KeyStrokes als global registriert werden können. Ebenfalls soll der Widget-Steuerung, welche auf dem UI implementiert ist eine Modifier-Taste hinzugefügt werden.

Dazu gibt es zwei unterschiedliche Lösungsmöglichkeiten:

* Diese Informationen werden Auf dem Desktop abgelegt. ->ein Widget registriert seine KeyStrokes als global verfügbar. Optional kann eine Modifier-Taste angeben.
* Die Informationen über Modifier-Taste und global verfügbar werden auf dem Model-Widget z.B. Table geführt.

Für die erste der beiden Möglichkeiten spricht, dass die KeyStrokes schlussendlich rein logisch auf dem Desktop registriert werden, auch wenn diese nicht auf der Modelseite(Widgetsteuerungen sind direkt auf dem UI implementiert.) existieren.

1. Für den Punkt 6. wird der AbstractDesktop wie folgt erweitert:

* getConfiguredAutoViewKeyStroke-> schaltet automatische Verteilung von Nummern für hinzugefügte Views ein oder aus.
* Logik beim Hinzufügen und Entfernen von Views, für das automatische Verteilen von Nummer-KeyStrokes für die einzelnen Views.

1. Für Punkt 7. wird nichts angepasst. ProcessButtons können mit dem & im Labeltext mit KeyStrokes versehen werden. Diese Keystrokes für ProcessButtons sind nur verfügbar wenn der Fokus sich im gleichen Form wie die Buttons befindet.
2. Mnemonics und DefaultButtons werden immer auf dem nächst höheren Form bzw. wenn kein Form vorhanden ist auf dem Desktop registriert.

## Priorisierung der Keystrokes

* Globale KeyStrokes gewinnen immer.
* Ist der KeyStroke nicht auf dem Desktop registriert, gewinnt der unterste in der Eventhierarchie. In folgendem Beispiel wird beim Betätigen von CTRL+T die Aktion des KeyStroke auf der Groupbox1 ausgeführt.: