App.js	
(function())	Lance le code se trouvant à l'intérieur
Todo()	Fonction qui instancie les différents fichiers Js
var todo	Instancie une nouvelle Todo
setView()	Ajoute la bonne url dans la page active

Controller.js	
(function())	Lance le code se trouvant à l'intérieur
Controller()	Constructeur (model, vue)
Controller.prototype.setView ()	Charge et initialise la vue
Controller.prototype.showAll()	Affiche tout les éléments dans la Todo-list
Controller.prototype,showActive()	Rends toutes les taches actives
Controller.prototype.showCompleted ()	Rends toutes les taches terminées
Controller.prototype.addItem ()	Ajout d'un élément (title)
Controller.prototype.editItem()	Déclenche le mode d'édition d'élements
Controller.prototype.editItemSave()	Sauvegarde l'édition de l'éléments (id,title)
Controller.prototype.editItemCancel()	Annule le mode édition d'élément
Controller.prototype.removeItem()	Supprime la tache de l'id passé en parametre (id)
prototype.removeCompletedItem()	Supprime tout les éléments terminé du dom et du storage
Controller.prototype.toggleComplete()	Affiche les todos selons leurs états (id, completed,silent)
Controller.prototype.toggleAll()	Avtive/Désactive les cases à cocher (completed)
Controller.prototype._updateCount()	Met à jour la page en fonction du nombre de todo restant
Controller.prototype._filter()	Filtre les elem de la todo en fonction de l'itineraire actif
Controller.prototype._updateFilterState()	Met à jour les état sélectionnés en fonction du filtre nav

Helper.js	
(function())	Lance le code se trouvant à l'intérieur
qs()	QuerySelector()
qsa()	QuerySelectorAll()
$on()	addEventListener Wrapper
$delegate()	associe un getionnaire d'event aux elem qui corresponde
$parent()	Rechercle le parent d'un élément via le tagName

Model.js	
(function())	Lance le code se trouvant à l'intérieur
Model()	Constructeur (storage)
Model.prototype.create()	Crée un nouveau modele de Todo-list (title,callback)
Model.prototype.read()	Trouve et renvoie un modele en mémoire (query,callback)
Model.prototype.update()	Met a jour le modele (id,data,callback)
Model.prototype.remove()	Supprime un modele du stockage (id, callback)
Model.prototype.removeAll()	Retire toutes les donnée du stockage (callback)
Model.prototype.getCount()	Renvoie le nombre de todos (callback)

Store.js	
(function())	Lance le code se trouvant à l'intérieur
Store()	Constructeur (name, callback)
Store.prototype.find()	Trouve les donnée correspondant (query,callback)
Store.prototype.findAll	Récupere toute les donnés (callback)
Store.prototype.save()	Sauvegarde les informations (updateDate, callback, id)
Store.prototype.remove()	Supprime un élément en fonction de son id (id, callback)
Store.prototype.drop()	Nouvel espace de stockage (callback)

Template.js	
(function())	Lance le code se trouvant à l'intérieur
htmlEscape	Constructeur (model, vue)
escapeHtmlChar	Regex
reUnescapedHtml	Regex
escape	Regex
template()	constructeur
template.prototype.show()	création d'un template (data)
template.prototype.itemCounter()	Affiche un compteur du nombre de tache a terminer
template.prototype.ClearCompledButton()	Affiche le bouton pour supprimer les tache terminé

View.js	
(function())	Lance le code se trouvant à l'intérieur
View()	Constructeur (model, vue)
Viewr.prototype._removeItem()	Supprime la lTodo en fonction de l'id
View.prototype._clearCompletedButton()	Masque les éléments completed
View.prototype._setFilter()	Affiche l'url courante
View.prototype.elementCompleted()	Test si la Todo est terminé (id, completed)
View.prototype._editItem()	edit la Todo (id, title)
View.prototype._editItemDone()	Remplace l'ancien élément par l'élement édité
View.prototype.render()	Affiche le rendu (viewCmd, parameter)
View.prototype._itemId()	Ajoute un ID à l'élément
View.prototype._bindItemEditDone()	Event sur la validation de l'édition d'un élément
_binditemEditCancel()	Event sur l'annulation de l'édition d'un élément
View.prototype.bind()	Fait le lien entre différentes méthodes