Helper.js	
qs()	querySelector
qsa()	querySelectorAll
$on()	encapsuler addEventListener
$delegate()	Délègue un eventListener à un parent
$parent()	recherche parent d' un élément via tagName

App.js	
Todo()	nouvelle app grace appel à storage / model / template / view / controller
var todo	lance la fonction Todo()
setView()	ajoute la route de la page active dans l' url
$on()	target : window, type : load, callback : setView
$on()	target : window, type : hasChange, callback : setView

Store.js	
Store()	(name, callback) constructor
Store.prototype.find()	(query, callback) Trouve les données correspondant à une requête
Store.prototype.findAll()	(callback) Récupère toutes les données
Store.prototype.save()	(updateData, callback, id) sauvegarde les données
Store.prototype.remove()	(id, callback) Enlève un élément en fonction de son ID
Store.prototype.drop()	(callback) Nouveau storage

Model.js	
Model()	(storage) constructor
Model.prototype.create()	(title, callback) nouveau model de todo
Model.prototype.read()	(query, callback) Trouve et renvoie un modèle en mémoire
Model.prototype.update()	(id, data, callback) Mise à jour
Model.prototype.remove()	(id, callback) Supprime les données correspondant à un ID
Model.prototype.removeAll()	(callback) Supprime toutes les données
Model.prototype.getCount()	(callback) Compteur

Template.js	
htmlEscapes	regex
escapeHtmlChar	regex
reUnescapedHtml	regex
escape	regex
Template()	Template HTML constructor
Template.prototype.show()	(data) Création du template
Template.prototype.itemCounter()	(activeTodos) Affiche un compteur
Template.prototype.clearCompletedButton()	(completedTodos) Affiche un bouton pour effacer

View.js	
View()	(template) constructor
View.prototype._removeItem()	(id) Supprime la Todo en fonction de l' id
View.prototype._clearCompletedButton()	Masque les éléments terminés
View.prototype._setFilter()	Indique la page actuelle
View.prototype._elementComplete()	(id, completed) test si le todo est terminé
View.prototype._editItem()	(id, title) édité le todo
iew.prototype._editItemDone()	Remplace l' ancien élément par l' élément édité
View.prototype.render()	(viewCmd, parameter) Affiche le render
View.prototype._itemId()	Ajoute un ID à l' élément
View.prototype._bindItemEditDone()	EventListener sur la validation de l' édition d'un élément
_bindItemEditCancel()	EventListener sur l' annulation de l' édition d'un élément
View.prototype.bind()	Fait le lien entre les méthodes du controller (controller.js) et les éléments de view (view.js)

Controller.js	
Controller()	(model, view) constructor
Controller.prototype.setView()	(locationHash) charge setView
Controller.prototype.showAll()	Affiche tous les éléments dans l' appli todo
Controller.prototype.showActive()	Affiche les todos en cours
Controller.prototype.showCompleted()	Affiche les todos terminés
Controller.prototype.addItem()	(title) Ajout d' élément
Controller.prototype.editItem()	(id) Permet l' édit de todo
Controller.prototype.editItemSave()	(id, title) sauvegarde l' édition
Controller.prototype.removeItem()	(id) supprime le todo corresponda à l' id passé en paramètre
Controller.prototype.removeCompletedItems()	supprime tous les todos terminés
Controller.prototype.toggleComplete()	(id, completed, silent) affiche les todos suivant leur état
Controller.prototype.toggleAll()	(completed) montre tous les todos
Controller.prototype._updateCount()	compteur
Controller.prototype._filter()	Filtre les éléments de la todo en fonction de l'itinéraire actif.
Controller.prototype._updateFilterState()	Met à jour les routes dans url