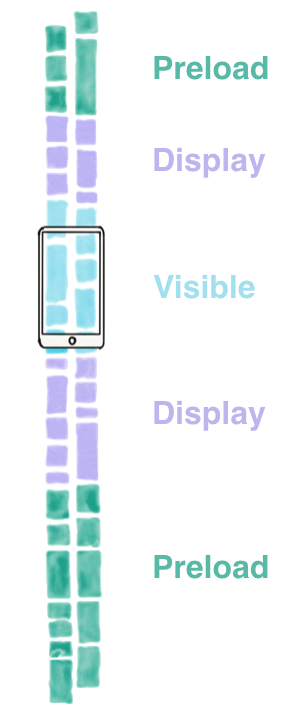
Texture (AsyncDisplayKit)

* Phân chia công việc chia công việc cho background thead để xử lý, chuẩn bị trước cho main thread để xử lý.
* Đơn vị cơ bản là Node (ASDisplayNode)



* Với giao diện cỏ thể scroll giao diện được chia làm 3 trạng thái
* Preload: nơi xa nhật trước khi hiển thị, nơi thu thập dữ liệu từ server hoặc thao tác với dữ liệu
* Display: thực hiện tác vụ xử lý giải mã ảnh, text,…
* Visible: Node được hiển thị trên màn hình
* Ta có thể custom độ dài của trạng thái Preload và display (tính theo màn hình; vd: display có độ dài 2 màn hình), khi tăng độ dài lên thì dung lượng cũng tăng

2. layout

* Layout được xây dựng trên horizontal và vertical layout.
* Một số loại layout đặc biệt được base trên Layout Spect và layout elements:
* ASStackLayoutSpec
* ASInsetLayoutSpec
* ASOverlayLayoutSpec
* ASAbsoluteLayoutSpec

3. hit test slop

Tăng khoảng tương tác của TextNode:

ASTextNode \*textNode = [[ASTextNode alloc] init];

CGFloat padding = (44.0 - button.bounds.size.height)/2.0;

textNode.hitTestSlop = UIEdgeInsetsMake(-padding, 0, -padding, 0);

4. layer backing

- để tương tác với một layer thì cần có 1 view nằm trên nó. Để bỏ các view đối với layer không tương tác thì bật

rootNode.isLayerBacked = YES;

MTProto mobile Protocal (<https://core.telegram.org/mtproto>)

* Được thiết kế để truy cập đến server API từ ứng dụng di dộng
* Protocol được chia làm 3 components độc lập
* **High-level component:** định nghĩa phương thức, theo đó query và response API sẽ được biến đổi thành binary message.
* **Cryptograpthic (authorization) layer:** định nghĩa phương thức mà từ đó message được mã hoá trước khi truyền qua transport protocol.
* Transport component: định nghĩa phương thức cho clien và server truyền message qua các giao thức mạng (vd: HTTP, HTTPS, WS (plain websockets), WSS (websockets over HTTPS), TCP, UDP)

1. Hight-level component (RPC Query language/ API)

* Server và client trao đổi message bên trong 1 session. Session đó được đính kèm với thiết bị khách thay vì liên kết websocket/http/https/tcp. Mỗi session được đính kèm vào user key id which authorization is actually accomplished.
* Nhiều đến server có thể được mở, messages có thể được gửi theo cả 2 chiều thông qua nhiều liên kết (response từ 1 query không nhất thiết phải được trả về từ connection đã gửi query mà có thể trả về từ một session khác). Khi mà UDP protocol được sử dụng, response có thể được trả về từ 1 địa chỉ IP khác, thay vì địa chỉ IP query đã được gửi đến.
* Một vài kiểu message:
* RPC calls (client to server): calls to API methods.
* RPC responses (server to client): results of RPC calls.
* Message received acknowledgment (or rather, notification of status of a set of message)
* Message status query
* Multipart message or container (ví dụ: một container nhiều messages, cần được gửi nhiều RPC calls một lúc qua kết nốt HTTP, container có thể hỗ trợ gzip)
* 1 message là 1 dữ liệu nhị phân 4 hoặc 16 byte. 1 vài trường đầu của message được fix cứng bởi hệ thống mã hoá.
* Mỗi message, độc lập hoặc message trong container, được cấu tạo bởi message identifier (64 bit), 1 message bao gồm nhiều 4 bytes. Mỗi khi container hoặc message độc lập được gửi, 1 internal header được thêm vào đầu trước khi cả đoạn message được mã hoá, và một external header được đặt vào đầu của message (64-bit key identifier và 128-bit message key)
* Một message body được tạo ra từ 32-bit message type theo sau là type-dependent parameters. Cụ thể, mỗi RPC function có 1 message type tương ứng.
* Tất cả number được viết dưới dạng little endian, very large numbers (2048-bit) sử dụng trong RSA và DH được viết trong big endian format do cơ chế của OpenSSL.

1. Authorization and Encryption

* Trước khi message được truyền qua network sử dụng transport protocal, nó được mã hoá theo một cách nhất định, và external header được thêm vào đầu của message là 64-kit key identifier (nó dùng để xác định authorization key cho server và user) và 128-bit message key. User key và message key tạo thành 256-bit key which is what encrypts the message using AES-256 encrytion. Phần đầu của message được mã hoá chứa nhiều loại dữ liệu (session, message ID, sequence number, server salt) nó chắc chắn sẽ ảnh hướng đến message key (và AES key và iv). Message key được định nghĩa như 128 middle bits SHA256 của message body (bao gồm session, message id, …) bao gồm cả padding bytes, được bổ sung bở 32 bytes lấy từ authorization key. Các mảnh của message được mã hoá như 1 message.